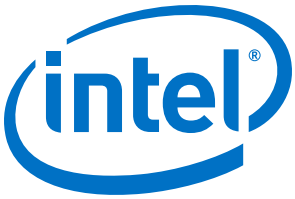


Intel® Stratix® 10 Clocking and PLL User Guide

Updated for Intel® Quartus® Prime Design Suite: **19.2**

UG-S10CLKPLL | 2019.07.01
Latest document on the web: **PDF** | **HTML**

Contents

1. Intel® Stratix® 10 Clocking and PLL Overview

1.1. Clock Networks Overview

Intel® Stratix® 10 devices contain dedicated resources for distributing signals throughout the fabric with balanced delay. These resources are typically used for clock signals and other signals with low-skew requirements. In Intel Stratix 10 devices, these resources are implemented as a programmable clock routing network, which allows for the implementation of low-skew clock trees of various size.

Related Information

Use Global Clock Network Resources, Design Recommendations User Guide (Intel Quartus® Prime Pro Edition)
Provides more information about clock assignments in the Intel Quartus® Prime software.

1.2. PLLs Overview

Phase-locked loops (PLLs) provide robust clock management and synthesis for device clock management, external system clock management, and high-speed I/O interfaces.

The Intel Stratix 10 device family contains the following PLLs for core applications:

- fPLLs—can function as fractional PLLs or integer PLLs
- I/O PLLs—can only function as integer PLLs

The fPLLs are located adjacent to the transceiver blocks in the transceiver banks. Each transceiver bank contains two fPLLs. You can configure each fPLL independently in either conventional integer mode, or fractional mode. In fractional mode, the fPLL can operate with third-order delta-sigma modulation. You can configure each fPLL to generate either a transmitter (TX) clock for a transceiver or to provide a single clock to the core.

The I/O PLLs are located adjacent to the hard memory controllers and LVDS serializer/deserializer (SERDES) blocks in the I/O banks. Each I/O bank contains one I/O PLL. The I/O PLLs can operate in conventional integer mode. Each I/O PLL has nine C counter outputs.

Intel Stratix 10 devices have up to 48 fPLLs and 42 I/O PLLs in the largest densities devices.

2. Intel Stratix 10 Clocking and PLL Architecture and Features

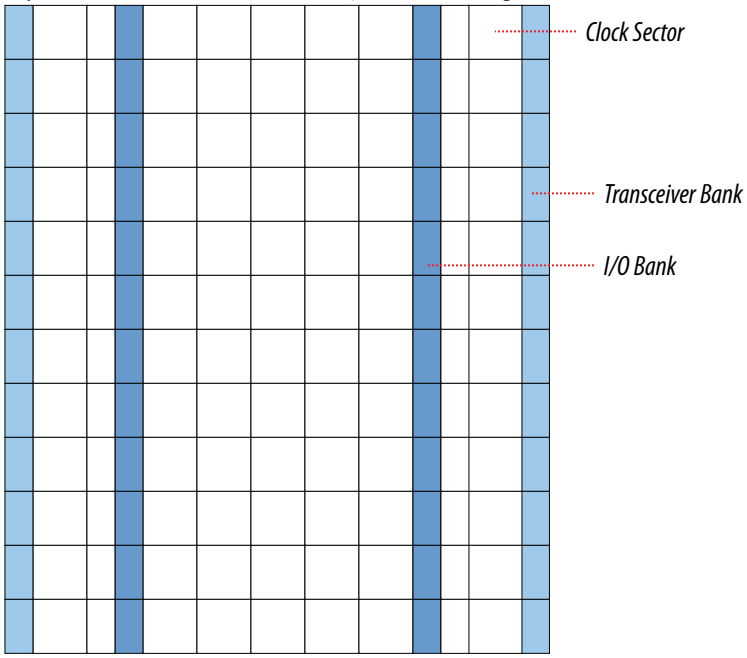
2.1. Clock Networks Architecture and Features

2.1.1. Clock Network Architecture

Each Intel Stratix 10 device is divided into a number of evenly sized clock sectors.

Figure 1. Clock Sector Floorplan for Intel Stratix 10 Devices

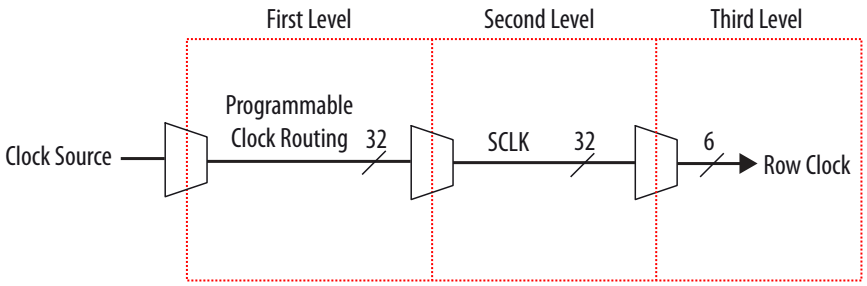
This figure shows an example of the clock sectors in an Intel Stratix 10 device, which is implemented as an array of sectors—12 rows and 9 columns in this example. Clock sectors are vertically aligned to match the height of transceiver and I/O banks. I/O banks are contained within the clock sectors. Transceiver bank interfaces are always located beside the clock sectors, at the left or right side of the device.



2.1.1.1. Clock Network Hierarchy

The Intel Stratix 10 clock network is organized in a hierarchy with 3 levels.

Figure 2. Clock Network Hierarchy

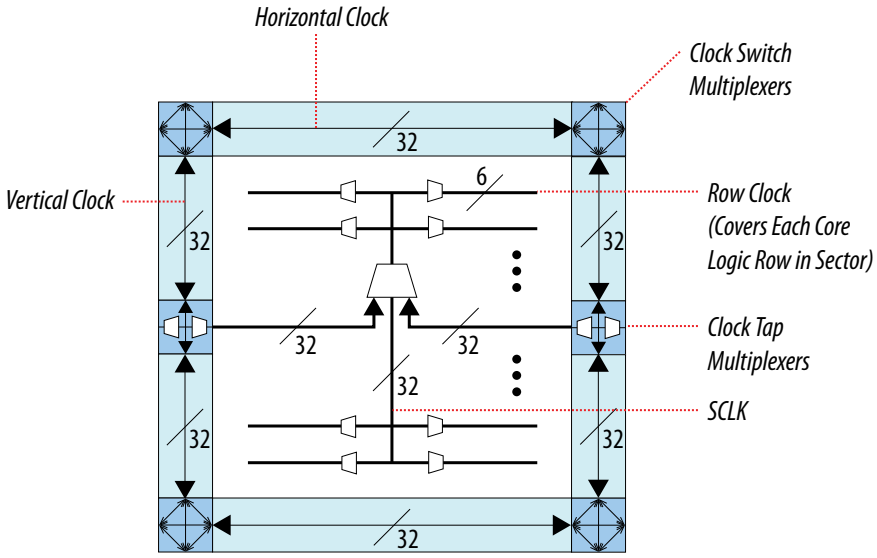


2.1.1.2. Clock Sector

Each clock sector has dedicated sector clock (SCLK) and row clock network resources that can be accessed by the programmable clock network. Each clock sector is also surrounded by programmable clock network resources. On each side, there is a channel that contains 32 independent bidirectional clock wires. At each corner, there is a set of programmable clock switch multiplexers which can route between these clocks wires.

A signal on a vertical clock wire can enter the sector to its left or right via clock tap multiplexers. The clock tap multiplexer drives a sector clock, which distributes the signal to each row in the clock sector. In each row, there are six row clock resources which connect to all core functional blocks, PLLs, and I/O interfaces in the sector, and to adjacent transceivers.

Figure 3. Dedicated Clock Resources Within a Clock Sector



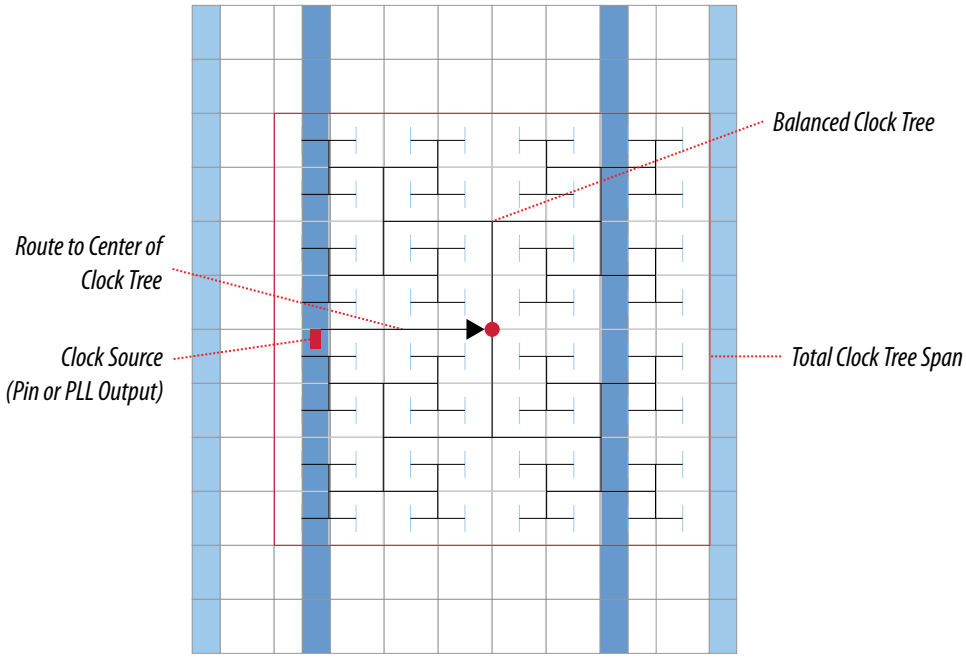
2.1.1.3. Programmable Clock Routing

The Intel Quartus® Prime software automatically configures the clock switch, clock tap, SCLK, and row clock multiplexers to generate skew-balanced clock trees. The resulting routing path distributes the signal from the clock source to all target destinations in one or more clock sectors.

The Intel Quartus Prime software creates efficiently balanced clock trees of various sizes, ranging from a single clock sector to the entire device, as shown in the following figure. By default, the Intel Quartus Prime Software automatically determines the size and location of the clock tree. Alternatively, you can directly constrain the clock tree size and location either with a Clock Region assignment or by Logic Lock Regions.

The total insertion delay for the clock network depends on the number of clock resources needed to implement the clock tree, increasing with the number of clock sectors reached and the distance of the furthest clock destination from the signal source. As delay increases, the worst-case skew for crossing clock sectors using different clock tree branches grows, potentially degrading the maximum performance. For very high-speed clock signals, it is advantageous to reduce the number of clock sectors driven, which reduces the clock skew, and to reduce the distance between the clock source and the furthest destination, which reduces both clock skew and total clock insertion delay.

Figure 4. Examples of Clock Networks Sizes Using Intel Stratix 10 Programmable Clock Routing



2.1.2. Clock Resources

Table 1. Intel Stratix 10 Clock Input Pins Resources

Device	Number of Resources Available	Source of Clock Resource
• GX 400 • SX 400	Transceiver: 24 differential I/O: 32 single-ended or 16 differential	For Transceiver pins: REFCLK_GXB[L,R][1,4][C,D,E,F,G,H,I,J,K,L,M,N]_CH[B,T][p,n] For I/O pins: CLK_[2,3][A..N]_[0,1][p,n]
• GX 650 • SX 650	Transceiver: 48 differential I/O: 32 single-ended or 16 differential	

continued...

Device	Number of Resources Available	Source of Clock Resource
• GX 850 • GX 1100 • SX 850 • SX 1100	Transceiver: 32 differential I/O: 60 single-ended or 30 differential	
• TX 850 • TX 1100	Transceiver: 26 differential I/O: 36 single-ended or 18 differential	
• GX 1650 • GX 2100 • SX 1650 • SX 2100	Transceiver: 32 differential I/O: 56 single-ended or 32 differential	
• MX 1650 • MX 2100	Transceiver: 32 differential I/O: 52 single-ended or 26 differential	
• TX 1650 • TX 2100	Transceiver: 35 differential I/O: 36 single-ended or 18 differential	
• GX 1660 • GX 2110	Transceiver: 16 differential I/O: 56 single-ended or 28 differential	
• GX 2500 • GX 2800 • SX 2500 • SX 2800	Transceiver: 32 differential I/O: 96 single-ended or 48 differential	
• TX 2500 • TX 2800	Transceiver: 53 differential I/O: 36 single-ended or 18 differential	

Table 2. Intel Stratix 10 Programmable Clock Routing Resources

Device	Number of Resources Available	Source of Clock Resource
All Intel Stratix 10 devices	32 bidirectional programmable clock routing at the boundary of each clock sector	For transceiver bank: • Physical medium attachment (PMA) and physical coding sublayer (PCS) TX and RX clocks per channel • PMA and PCS TX and RX divide clocks per channel • Hard IP core clock output signals • Fractional PLL (fPLL) C counter outputs • REFCLK pins • Core signals (1) For I/O bank: • I/O PLL C counter outputs • I/O PLL M counter outputs for feedback • Clock input pins • Core signals (1) • Dynamic phase alignment (DPA) clock output • Phase aligner counter outputs

For more information about the clock input pins connections, refer to the pin connection guidelines.

Related Information

Intel Stratix 10 Device Family Pin Connection Guidelines

(1) Core signals drive directly to programmable clock routing through clock switch multiplexers in the clock sector instead of the periphery DCM block.

2.1.3. Programmable Clock Routing Sources

This section describes the sources that can drive the programmable clock routing.

2.1.3.1. Dedicated Clock Input Pins

The sources of dedicated clock input pins are as follows:

- fPLL—REFCLK_GXB[L,R][1,4][C,D,E,F,G,H,I,J,K,L,M,N]_CH[B,T][p,n] from transceiver column
- I/O PLL—CLK_[2,3][A..N]_[0,1][p,n] from I/O column

You can use the dedicated clock input pins for high fan-out control signals, such as asynchronous clears, presets, and clock enables, for protocol signals through the programmable clock routing.

The dedicated clock input pins for an I/O PLL can be either differential clocks or single-ended clocks. The dedicated clock input pins for fPLL only support differential clocks and do not support single-ended clocks.

Driving a PLL over programmable clock routing can cause higher jitter at the PLL input, and the PLL is not able to fully compensate for the programmable clock routing. Intel recommends using the dedicated clock input pins for optimal performance to drive the PLLs.

2.1.3.2. Internal Logic

You can route up to eight core signals to each clock switch multiplexer, except the clock switch multiplexers at the right and left edge of the device, and the clock switch multiplexers next to the I/O banks.

2.1.3.3. DPA Clock Outputs

Each DPA clock output can drive the programmable clock routing.

2.1.3.4. Transceiver Clock Outputs

PMA and PCS TX and RX clock outputs can drive the programmable clock routing.

2.1.3.5. PLL Clock Outputs

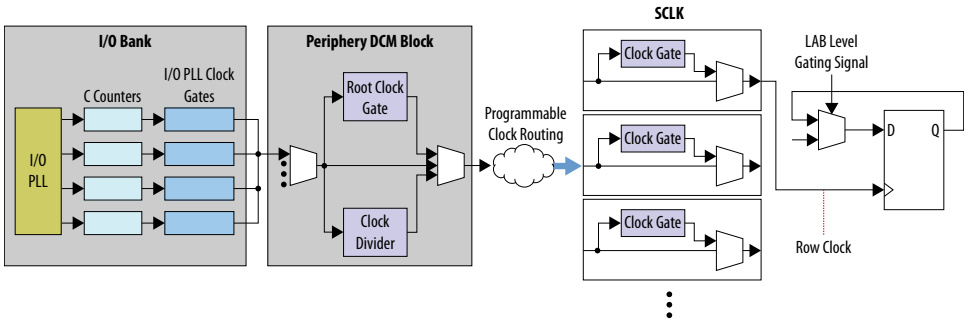
The fPLL and I/O PLL clock outputs can drive the programmable clock routing.

2.1.4. Clock Control Features

The following figure shows the high level description of the Intel Stratix 10 clock control features—clock gating and clock divider. The clock from the I/O PLL output can be gated dynamically. These clock signals along with other clock sources go to the periphery distributed clock multiplexer (DCM). In the periphery DCM, the clock signal can either pass straight through, be gated by the root clock gate, or be divided by the clock divider.

The Intel Quartus Prime software routes the clock signal on the programmable clock routing to reach each clock sector. The clock signal can be gated in each sector by the SCLK gates. The clock enters the SCLK network followed by the row clock network, and eventually reaches the registers in the core. The LAB registers have a built-in functional clock enable feature, as shown in the following figure.

Figure 5. Clock Gating and Clock Divider in Intel Stratix 10 Clock Network



2.1.4.1. Clock Gating

2.1.4.1.1. Root Clock Gate

There is one root clock gate per I/O bank and transceiver bank. This gate is a part of the periphery DCM and is located close to the clock buffer.

The Intel Stratix 10 root clock gate is intended for limited clock gating scenarios where high insertion delay can be tolerated. When you use a root clock gate, set `multicycle` of several clock cycles between the generation of the clock gating signal in the core and the gated clock in the periphery to meet the timing requirement. For high frequency clocks that require single-cycle gating, use sector clock gates.

Related Information

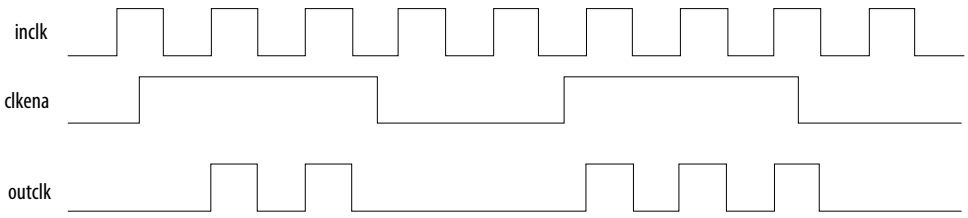
Clock Control IP Core Parameters on page 47
Select **Clock Enable Type ➤ Root Level** in the Clock Control IP core.

2.1.4.1.2. Sector Clock Gate

There are 32 SCLKs in every sector of the device. Each SCLK has a clock gate and bypassable clock gate path. The SCLK gates are controlled by clock enable inputs from the core logic. The Intel Quartus Prime software can route up to eight unique clock enable signals to the 32 SCLKs in a sector.

Intel recommends using the clock gate with a negative latch to provide glitch free gating on the output clock signal (`outclk`). The clock gate captures the enable signal (`clkena`) on the next rising edge of the input clock signal (`inclk`). The following timing diagram shows the relationship of the `outclk` with respect to `inclk` and `clkena`.

Figure 6. Clock Gating Timing Diagram



The clock signal going into the SCLK network in a sector can only reach the core logic in that sector. When you instantiate a SCLK gate in your design, the Intel Quartus Prime software automatically duplicates the SCLK gate to create a clock gate in every sector to which the clock signal is routed.

The SCLK gate is suitable for cycle-specific clock gating for high-frequency clocks. The timing of the enable path to the SCLK gate is analyzed by the Intel Quartus Prime software.

Related Information

- Clock Sector on page 6
  Provides a diagram that shows the dedicated clock resources within a clock sector.
- Clock Control Features on page 9
  Provides a diagram that shows the resources within a SCLK.
- Clock Control IP Core Parameters on page 47
  Select **Clock Enable Type ➤ Distributed Sector Level** in the Clock Control IP core.

2.1.4.1.3. I/O PLL Clock Gate

You can dynamically gate each output counter of the Intel Stratix 10 I/O PLL. This provides a useful alternative to the root clock gate because the root clock gate can gate only 1 of the 9 output counters.

However, the I/O PLL clock gate is not cycle-specific. When you use the I/O PLL clock gate, expect a delay of several clock cycles between the assertion or deassertion of the clock gate and the corresponding change to the clock signal. The number of delay cycles is non-deterministic because the enable signal must be synchronized into the clock domain of the output clock, ensuring a glitch-free gate.

2.1.4.1.4. LAB Clock Gate

The Intel Stratix 10 LAB register has built-in clock gating functionality. The register clock enable mechanism is a hardened data feedback, as shown in the Clock Gating and Clock Divider in Intel Stratix 10 Clock Network diagram. The LAB clock gate offers no associated power savings because this is a purely functional clock enable.

The analysis and synthesis phases of the Intel Quartus Prime software infer a LAB clock gate from a behavioral description of clock gating in the register transfer level (RTL). If a physical clock gate is desired, you must instantiate it explicitly.

Related Information

Clock Control Features on page 9
Provides the Clock Gating and Clock Divider in Intel Stratix 10 Clock Network diagram.

2.1.4.2. Clock Divider

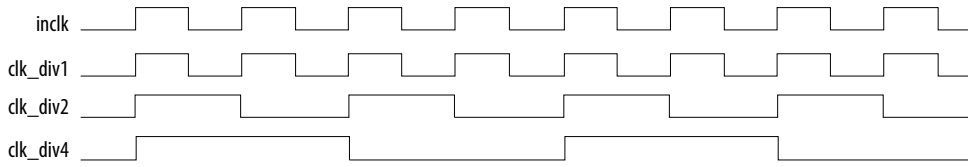
There is one clock divider per I/O bank and transceiver bank. The clock divider is a part of the periphery DCM block and is located close to the root clock gate. The outputs of the clock divider cannot be gated by the root clock gate in the same periphery DCM block. However, this limitation does not apply to the SCLK gate. The clock divider output in the periphery DCM block can drive a SCLK gate after going through the programmable clock routing.

The clock divider has three outputs as follows:

- First output—Passes through the input clock.
- Second output—Divides the input clock by two.
- Third output—Divides the input clock by four.

These three clock outputs are edge-aligned at the output of the clock divider.

Figure 7. Clock Divider Timing Diagram



Related Information

Clock Control Features on page 9
Provides a diagram that shows the root clock gate and clock divider in the periphery DCM block.

2.1.4.3. Dynamic Clock Switchover

Intel Stratix 10 devices do not have hard clock multiplexer blocks for dynamic clock switchover. Thus, the dynamic clock switchover logic is implemented using the soft logic in the core. Optionally, you can make the dynamic clock switchover glitch free using additional external soft logic.

2.2. PLLs Architecture and Features

2.2.1. PLL Features

Table 3. PLL Features in Intel Stratix 10 Devices—Preliminary

Feature	Fractional PLL	I/O PLL
Integer PLL	Yes	Yes
Fractional PLL	Yes	—

continued...

Feature	Fractional PLL	I/O PLL
Number of C output counter	1	9
M counter divide factor range	In integer mode: 8 to 127 In fractional mode: 11 to 123	4 to 160
N counter divide factor range	1 to 32	1 to 110
C counter divide factor range	1 to 512	1 to 510
L counter divide factors	1, 2, 4, and 8	—
Dedicated external clock outputs	—	Yes
Dedicated clock input pins	Yes	Yes
External feedback input pin	—	Yes
Spread-spectrum input clock tracking (2)	Yes	Yes
Source synchronous compensation (3)	—	Yes
Direct compensation	Yes	Yes
Normal compensation (3)	—	Yes
Zero-delay buffer compensation	—	Yes
External feedback compensation	—	Yes
LVDS compensation	—	Yes
Voltage-controlled oscillator (VCO) output drives the DPA clock	—	Yes
Phase shift resolution (4)	—	78.125 ps
Programmable duty cycle	Fixed 50% duty cycle	Yes
Power down mode	Yes	Yes

2.2.2. PLL Usage

fPLLs are optimized for use as transceiver transmit PLLs and for synthesizing reference clock frequencies. You can use the fPLLs to:

- Transmit clocking for transceivers
- Reduce the number of required oscillators on the board

(2) Provided input clock jitter is within input jitter tolerance specifications.

(3) Non-dedicated feedback path option is available for this compensation mode.

(4) The smallest phase shift is determined by the VCO period divided by eight (for I/O PLL). For degree increments, the Intel Stratix 10 device can shift all output frequencies in increments of at least 45° (for I/O PLL). Smaller degree increments are possible depending on the frequency and divide parameters.

I/O PLLs are optimized for use with memory interfaces and LVDS SERDES. You can use the I/O PLLs to:

- Reduce the number of required oscillators on the board
- Reduce the clock pins used in the FPGA by synthesizing multiple clock frequencies from a single reference clock source
- Simplify the design of external memory interfaces and high-speed LVDS interfaces
- Ease timing closure because the I/O PLLs are tightly coupled with the I/Os
- Compensate for clock network delay
- Zero delay buffering

2.2.3. PLL Architecture

Figure 8. Fractional PLL High-Level Block Diagram for Intel Stratix 10 Devices

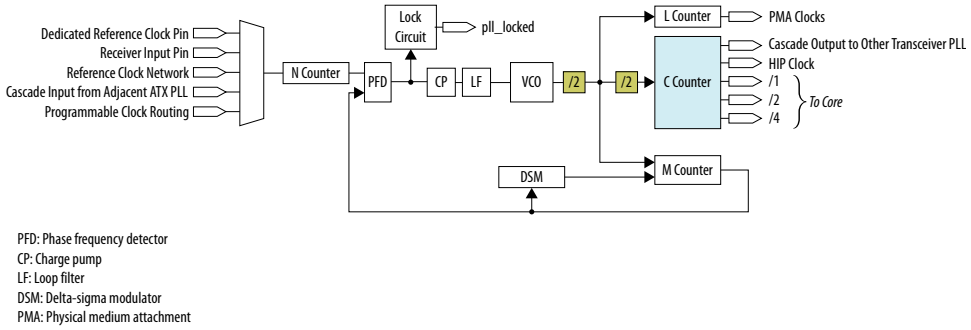
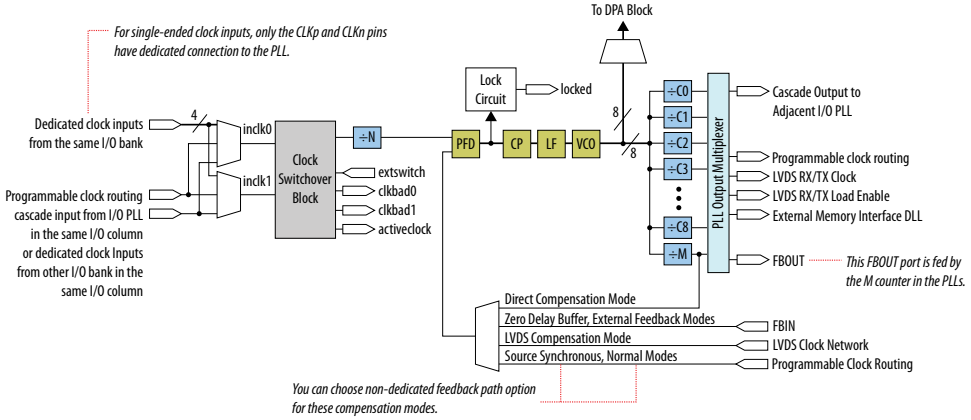


Figure 9. I/O PLL High-Level Block Diagram for Intel Stratix 10 Devices



2.2.4. PLL Control Signals

You can use the reset signal to control PLL operation and resynchronization, and use the locked signal to observe the status of the PLL.

2.2.4.1. Reset

The reset signal port of the IP core for I/O PLL is `reset`.

The reset signal is the reset or resynchronization input for each I/O PLL. The device input pins or internal logic can drive these input signals.

When the reset signal is driven high, the I/O PLL counters reset, clearing the I/O PLL output and placing the I/O PLL out-of-lock. The VCO is then set back to its nominal setting. When the reset signal is driven low again, the I/O PLL resynchronizes to its input clock source as it re-locks.

You must assert the reset signal every time the I/O PLL loses lock to guarantee the correct phase relationship between the I/O PLL input and output clocks. You can set up the I/O PLL to automatically reset (self-reset) after a loss-of-lock condition using the Intel Quartus Prime parameter editor.

You must include the reset signal if either of the following conditions is true:

- I/O PLL reconfiguration or clock switchover is enabled in the design.
- Phase relationships between the I/O PLL input and output clocks must be maintained after a loss-of-lock condition.

Note: Reset the I/O PLL after the input clock is stable and within specifications, even when the self-reset feature is enabled, if either one of the following conditions occur:

- The input clock to the I/O PLL is not toggling or is unstable when the FPGA transitions into user mode.
- The I/O PLL is not able to lock to the reference clock after reconfiguring the I/O PLL.

Related Information

PLL Calibration on page 29

2.2.4.2. Locked

The locked signal port of the IP core for each PLL is as follows:

- fPLL—`pll_locked`
- I/O PLL—`locked`

The lock detection circuit provides a signal to the core logic. The signal indicates when the feedback clock has locked onto the reference clock both in phase and frequency.

When PLL loses lock, the output of the PLL starts drifting out of the desired frequency. The downstream logic must be held inactive once PLL has lost lock.

2.2.5. Clock Feedback Modes

Clock feedback modes compensate for clock network delays to align the rising edge of the output clock with the rising edge of the PLL's reference clock. Select the appropriate type of compensation for the timing critical clock path in your design.

PLL compensation is not always needed. A PLL should be configured in direct (no compensation) mode unless a need for compensation is identified. Direct mode provides the best PLL jitter performance and avoids expending compensation clocking resources unnecessarily.

The default clock feedback mode is direct compensation mode.

fPLLs support only the direct compensation mode.

I/O PLLs support the following clock feedback modes:

- Direct compensation
- LVDS compensation
- Source synchronous compensation
- Normal compensation
- Zero delay buffer (ZDB) compensation
- External feedback (EFB) compensation

Normal and source synchronous compensation modes compensate for the insertion delay of a routed core clock. For Intel Stratix 10 devices, you can achieve core clock compensation by the following methods:

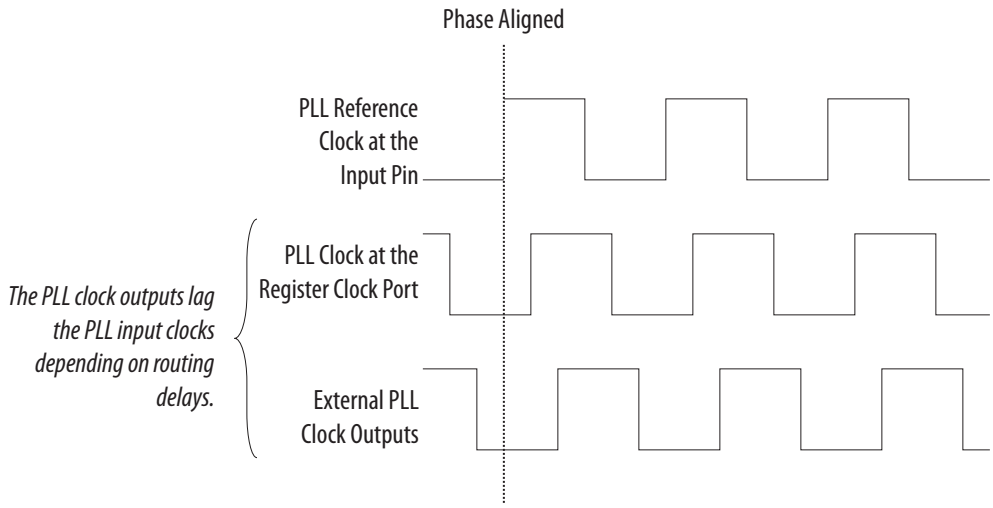
- You can route a dedicated feedback clock from the `M` counter in the I/O PLL to emulate the insertion delay of the compensated `C` counter output clock network.
- You can select **Use Nondedicated Feedback Path** from the IOPLL Intel FPGA IP core which routes the compensated `C` counter output clock back to the I/O PLL.

Intel recommends the non-dedicated feedback mechanism because it utilizes the clock resources most efficiently. The default is dedicated feedback when you choose normal or source synchronous compensation mode in the IOPLL IP core.

2.2.5.1. Direct Compensation Mode

In direct mode, the PLL does not compensate for any clock network delays. This mode provides better jitter performance compared to other compensation modes because the clock feedback into the phase frequency detector (PFD) passes through less circuitry. Both the PLL internal- and external-clock outputs are phase-shifted with respect to the PLL clock input.

Figure 10. Example of Phase Relationship Between the PLL Clocks in Direct Mode



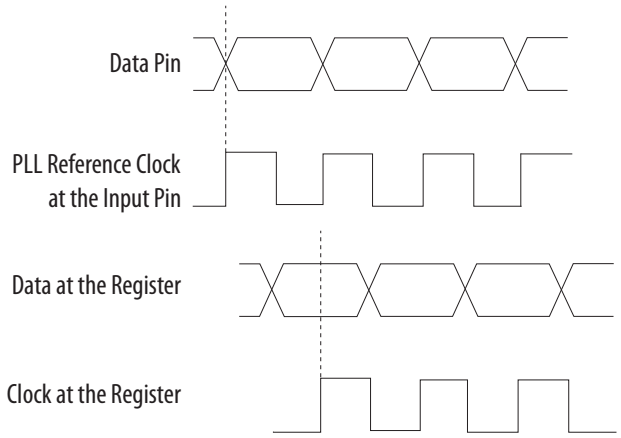
2.2.5.2. LVDS Compensation Mode

The purpose of LVDS compensation mode is to maintain the same data and clock timing relationship seen at the pins of the internal serializer/deserializer (SERDES) capture register, except that the clock is inverted (180° phase shift). Thus, LVDS compensation mode ideally compensates for the delay of the LVDS clock network, including the difference in delay between the following two paths:

- Data pin-to-SERDES capture register
- Clock input pin-to-SERDES capture register

The output counter must provide the 180° phase shift.

Figure 11. Example of Phase Relationship Between the Clock and Data in LVDS Compensation Mode

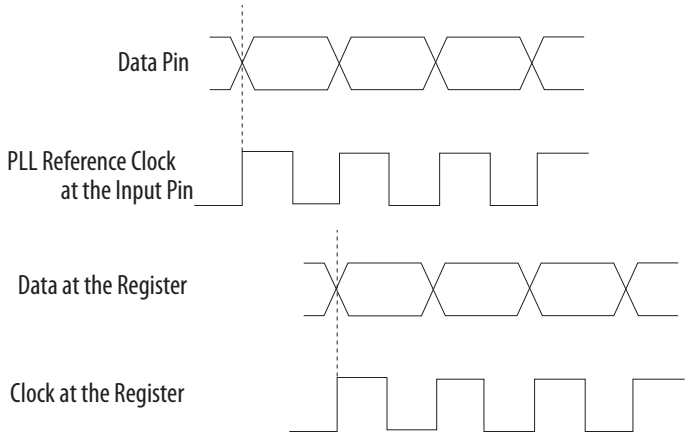


2.2.5.3. Source Synchronous Compensation Mode

If the data and clock signals arrive at the same time on the input pins, the same phase relationship is maintained at the clock and data ports of any IOE input register. Data and clock signals at the IOE experience similar buffer delays as long as you use the same I/O standard. Only one output clock can be compensated in source synchronous compensation mode.

Intel recommends source synchronous mode for source synchronous data transfers.

Figure 12. Example of Phase Relationship Between Clock and Data in Source Synchronous Mode



The source synchronous mode compensates for the delay of the clock network used and any difference in the delay between the following two paths:

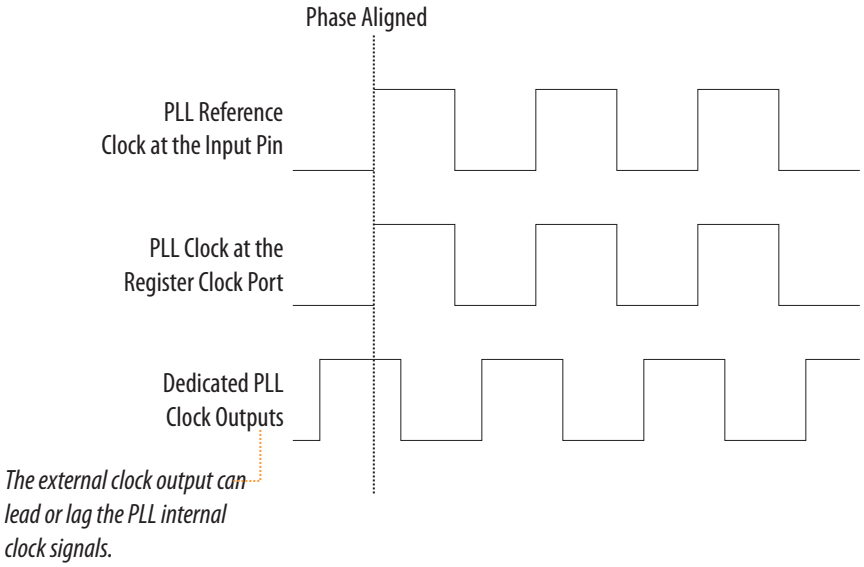
- Data pin to the IOE register input
- Clock input pin to the PLL PFD input

The Intel Stratix 10 PLL can compensate multiple pad-to-input-register paths, such as a data bus when it is set to use source synchronous compensation mode.

2.2.5.4. Normal Compensation Mode

An internal clock in normal compensation mode is phase-aligned to the input clock pin. The external clock output pin has a phase delay relative to the clock input pin if connected in this mode. The Intel Quartus Prime Timing Analyzer reports any phase difference between the two. In normal compensation mode, the delay introduced by the clock network is fully compensated. Only one output clock can be compensated in normal compensation mode.

Figure 13. Example of Phase Relationship Between the PLL Clocks in Normal Compensation Mode



2.2.5.5. Zero-Delay Buffer Mode

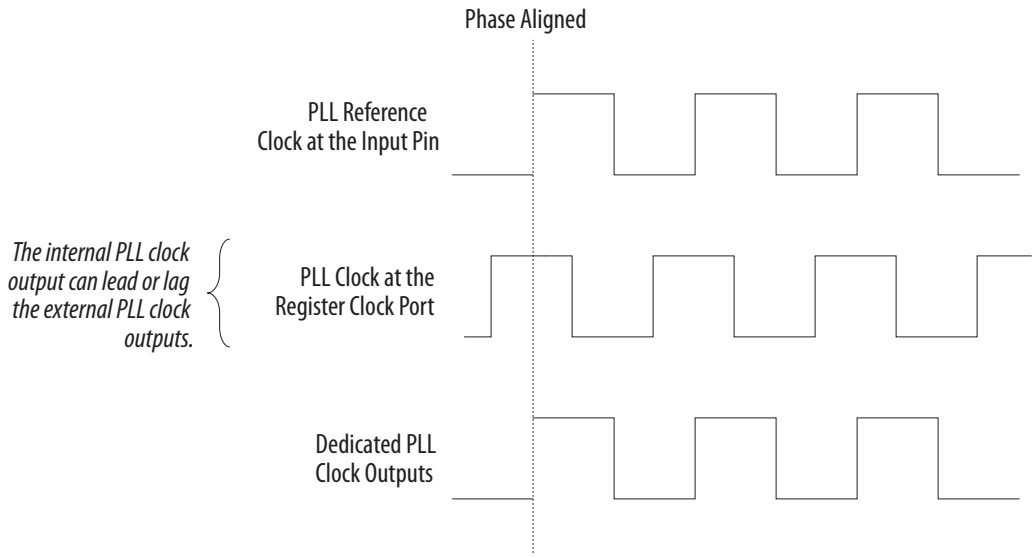
In zero-delay buffer (ZDB) mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device.

In this mode, you must use the same I/O standard on the input clocks and clock outputs to guarantee clock alignment at the input and output pins. You cannot use differential I/O standards on the PLL clock input or output pins.

To ensure phase alignment between the `clk` pin and the external clock output (CLKOUT) pin in ZDB mode, instantiate a bidirectional I/O pin in the design. The bidirectional I/O pin serves as the feedback path connecting the `fbout` and `fbin` ports of the PLL. The bidirectional I/O pin must always be assigned a single-ended I/O standard. The PLL uses this bidirectional I/O pin to mimic and compensate for the output delay from the clock output port of the PLL to the external clock output pin.

Note: To avoid signal reflection when using ZDB mode, do not place board traces on the bidirectional I/O pin.

Figure 14. Example of Phase Relationship Between the PLL Clocks in ZDB Mode



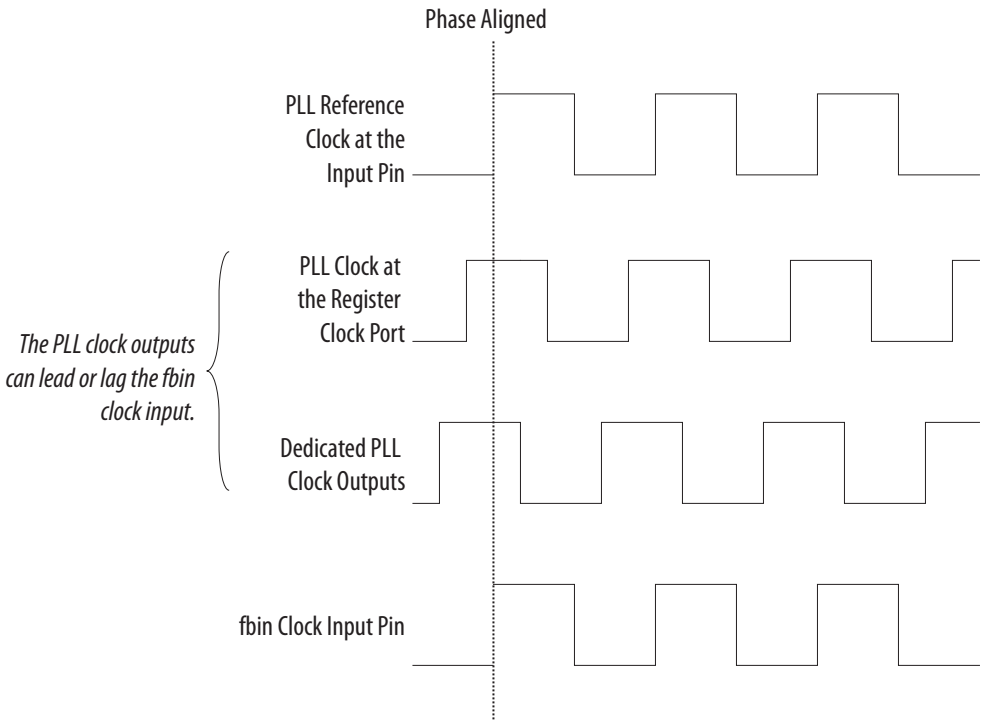
2.2.5.6. External Feedback Mode

In external feedback (EFB) mode, the output of the `M` counter (`fbout`) feeds back to the PLL `fbin` input (using a trace on the board) and becomes part of the feedback loop.

One of the dual-purpose external clock outputs becomes the `fbin` input pin in this mode. The external feedback input pin, `fbin` is phase-aligned with the clock input pin. Aligning these clocks allows you to remove clock delay and skew between devices.

In EFB mode, you must use the same I/O standard on the input clock, feedback input, and clock outputs.

Figure 15. Example of Phase Relationship Between the PLL Clocks in EFB Mode



2.2.6. Clock Multiplication and Division

An Intel Stratix 10 PLL output frequency is related to its input reference clock source by the following scale factors:

- M/(N × C) for I/O PLL
- M/(N × C × 2) for fPLL core applications

The input clock is divided by a pre-scale factor, N, and is then multiplied by the M feedback factor. The control loop drives the VCO to match f_{in} × (M/N). When using non-dedicated feedback path in normal or source synchronous compensation mode, the control loop drives the VCO to match f_{in} × ((M × C_i)/N), where C_i is the compensated `outclk` C counter value. The Intel Quartus Prime software automatically chooses the appropriate scale factors according to the input frequency, multiplication, and division values entered into the Intel FPGA IP cores for I/O PLL and fPLL.

Pre-Scale Counter, N and Multiply Counter, M

Each PLL has one pre-scale counter, N, and one multiply counter, M. The M and N counters do not use duty-cycle control because the only purpose of these counters is to calculate frequency division.

Post-Scale Counter, C

Each output port has a unique post-scale counter, C. For multiple C counter outputs with different frequencies, the VCO is set to the least common multiple of the output frequencies that meets its frequency specifications. For example, if the output

frequencies required from one I/O PLL are 55 MHz and 100 MHz, the Intel Quartus Prime software sets the VCO frequency to 1.1 GHz (the least common multiple of 55 MHz and 100 MHz within the VCO operating frequency range). Then the post-scale counters, C, scale down the VCO frequency for each output port.

Post-Scale Counter, L

The fPLL has an additional post-scale counter, L. The L counter synthesizes the frequency from its clock source using the M/(N × L) scale factor. The L counter generates a differential clock pair (0 degree and 180 degree) and drives the transceiver clock network.

Delta-Sigma Modulator

The delta-sigma modulator (DSM), together with the M multiply counter, enable the fPLL to operate in fractional mode. The DSM dynamically changes the M counter factor on a cycle-to-cycle basis. The changes in M counter factors result an average M counter factor that is non-integer.

Fractional Mode

In fractional mode, the M counter value equals the sum of the M feedback factor and the fractional value. The fractional value is equal to $K/2^X$, where K is an integer between 0 and $(2^X - 1)$, and $X = 32$.

Integer Mode

For a fPLL operating in integer mode, M is an integer value and DSM is disabled.

The I/O PLL can only operate in integer mode.

2.2.7. Programmable Phase Shift

The programmable phase shift feature allows only the I/O PLLs to generate output clocks with a fixed phase offset.

The VCO frequency of the PLL determines the precision of the phase shift. The minimum phase shift increment is 1/8 of the VCO period. For example, if an I/O PLL operates with a VCO frequency of 1000 MHz, phase shift steps of 125 ps are possible.

The Intel Quartus Prime software automatically adjusts the VCO frequency according to the user-specified phase shift values entered into the IOPLL IP core parameter editor.

2.2.8. Programmable Duty Cycle

The programmable duty cycle feature allows I/O PLLs to generate clock outputs with a variable duty cycle. This feature is only supported by the I/O PLL post-scale counters, C. fPLLs do not support the programmable duty cycle feature and only have fixed 50% duty cycle.

The I/O PLL C counter value determines the precision of the duty cycle. The precision is 50% divided by the post-scale counter value. For example, if the C0 counter is 10, steps of 5% are possible for duty-cycle options from 5% to 90%. If the I/O PLL is in external feedback mode, set the duty cycle for the counter driving the fbin pin to 50%.

The Intel Quartus Prime software automatically adjusts the VCO frequency according to the required duty cycle that you enter in the IOPLL IP core parameter editor.

Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

2.2.9. PLL Cascading

Intel Stratix 10 devices support PLL-to-PLL cascading. You can cascade a maximum of two PLLs. PLL cascading synthesizes more output clock frequencies than a single PLL.

If you cascade PLLs in your design, the source (upstream) PLL must have a low-bandwidth setting, and the destination (downstream) PLL must have a high-bandwidth setting for I/O PLL and medium-bandwidth setting for fPLL. During cascading, the output of the source PLL serves as the reference clock (input) of the destination PLL. The bandwidth settings of cascaded PLLs must be different. If the bandwidth settings of the cascaded PLLs are the same, the cascaded PLLs may amplify phase noise at certain frequencies.

Intel Stratix 10 devices support the following PLL-to-PLL cascading modes:

- I/O-PLL-to-I/O-PLL cascading via dedicated cascade path—Upstream I/O PLL and downstream I/O PLL must be in the same I/O column.
- I/O-PLL-to-I/O-PLL cascading via core clock fabric—No restriction on locations of upstream and downstream I/O PLL.

The permit_cal input of the downstream I/O PLL must be connected to the locked output of the upstream I/O PLL in both PLL cascading modes.

The following figures show the connectivity required between the upstream and downstream I/O PLL for both the PLL cascading modes.

Figure 16. I/O-PLL-to-I/O-PLL Cascading Via Dedicated Cascade Path

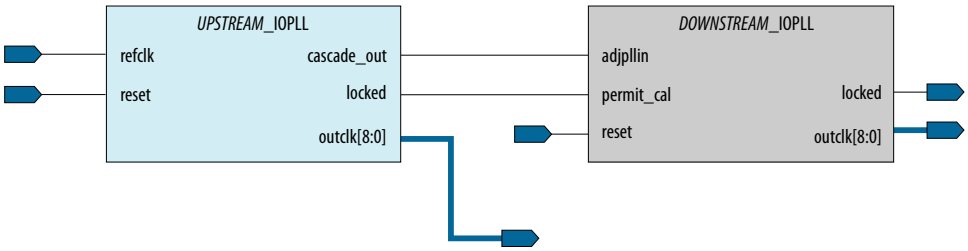
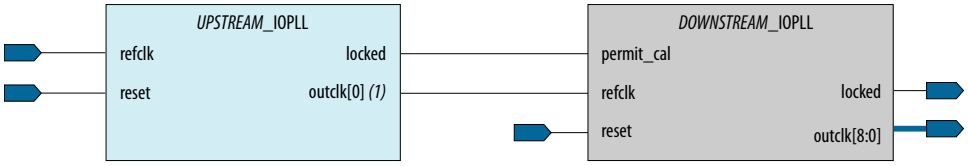


Figure 17. I/O-PLL-to-I/O-PLL Cascading Via Core Clock Fabric



Note:
(1) You may connect any of the outclk from the upstream I/O PLL to the refclk port in the downstream I/O PLL when cascading the I/O PLL via core clock fabric.

2.2.10. Clock Switchover

The clock switchover feature allows the I/O PLL to switch between two reference input clocks. Use this feature for clock redundancy or for a dual-clock domain application where a system turns to the redundant clock if the previous clock stops running. The design can perform clock switchover automatically when the clock is no longer toggling or based on a user control signal, `extswitch`.

Intel Stratix 10 I/O PLLs support the following clock switchover modes:

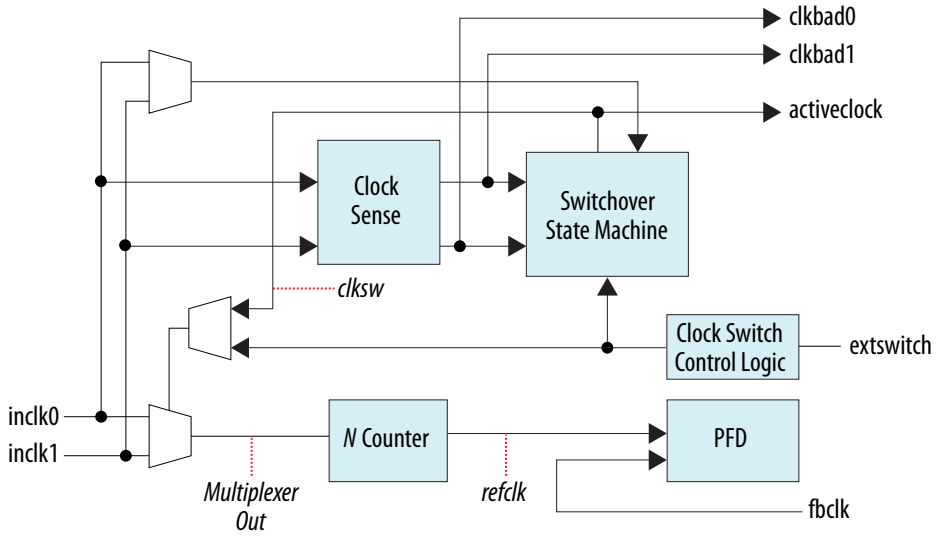
- Automatic switchover—The clock sense circuit monitors the current reference clock. If the current reference clock stops toggling, the reference clock automatically switches to `inclk0` or `inclk1` clock.
- Manual clock switchover—Clock switchover is controlled using the `extswitch` signal. When the `extswitch` signal goes from logic high to logic low, and stays low for at least three clock cycles for the `inclk` being switched to, the reference clock to the I/O PLL is switched from `inclk0` to `inclk1`, or vice-versa.
- Automatic switchover with manual override—This mode combines automatic switchover and manual clock switchover. When the `extswitch` signal goes low, it overrides the automatic clock switchover function. As long as the `extswitch` signal is low, further switchover action is blocked.

2.2.10.1. Automatic Switchover

Intel Stratix 10 I/O PLLs support a fully configurable clock switchover capability.

Figure 18. Automatic Clock Switchover Circuit Block Diagram

This figure shows a block diagram of the automatic switchover circuit built into the I/O PLL.



When the current reference clock is not present, the clock sense block automatically switches to the backup clock for I/O PLL reference. You can select a clock source as the backup clock by connecting it to the `inclk1` port of the I/O PLL in your design.

The clock switchover circuit sends out three status signals—`clkbad0`, `clkbad1`, and `activeclock`—from the I/O PLL to implement a custom switchover circuit in the logic array.

In automatic switchover mode, the `clkbad0` and `clkbad1` signals indicate the status of the two clock inputs. When they are asserted, the clock sense block detects that the corresponding clock input has stopped toggling. These two signals are not valid if the frequency difference between `inclk0` and `inclk1` is greater than 20%.

The `activeclock` signal indicates which of the two clock inputs (`inclk0` or `inclk1`) is being selected as the reference clock to the I/O PLL. When the frequency difference between the two clock inputs is more than 20%, the `activeclock` signal is the only valid status signal.

Use the switchover circuitry to automatically switch between `inclk0` and `inclk1` when the current reference clock to the I/O PLL stops toggling. You can switch back and forth between `inclk0` and `inclk1` any number of times when one of the two clocks fails and the other clock is available.

For example, in applications that require a redundant clock with the same frequency as the reference clock, the switchover state machine generates a signal (`clksw`) that controls the multiplexer select input. In this case, `inclk1` becomes the reference clock for the I/O PLL.

When using automatic clock switchover mode, the following requirements must be satisfied:

- Both clock inputs must be running when the FPGA is configured.
- The period of the two clock inputs can differ by no more than 20%.
- The input clocks must meet the input jitter specifications and I/O standard specifications.

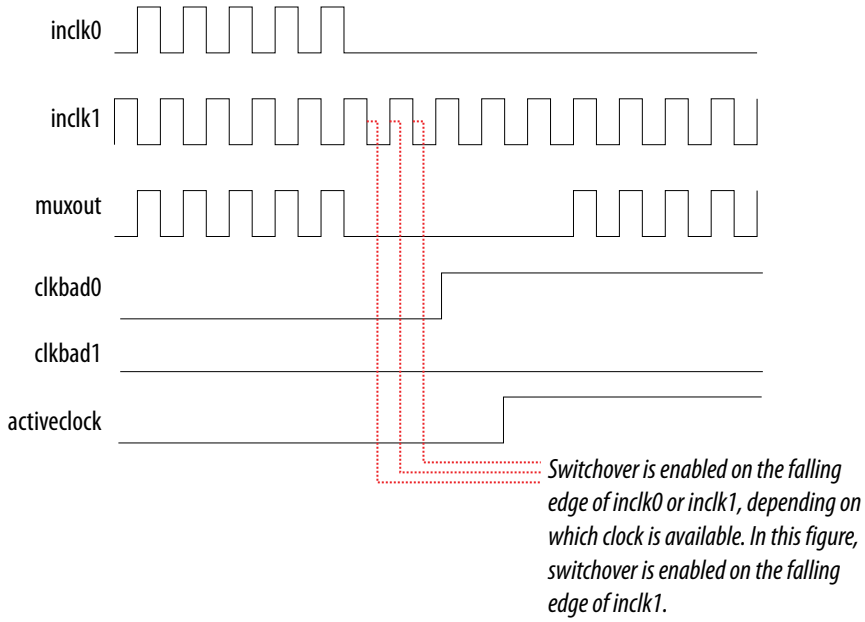
Glitches in the input clock may be seen as a greater than 20% difference in frequency between the input clocks.

If the current clock input stops toggling while the other clock is also not toggling, switchover is not initiated and the `clkbad[0..1]` signals are not valid. If both clock inputs are not the same frequency, but their period difference is within 20%, the clock sense block detects when a clock stops toggling. However, the I/O PLL may lose lock after the switchover is completed and needs time to relock.

Note: You must reset the I/O PLL using the reset signal to maintain the phase relationships between the I/O PLL input and output clocks when using clock switchover.

Figure 19. Automatic Switchover After Loss of Clock Detection

This figure shows an example waveform of the switchover feature in automatic switchover mode. In this example, the `inclk0` signal is held low. After the `inclk0` signal is held low for approximately two clock cycles, the clock sense circuitry drives the `clkbad0` signal high. As the reference clock signal (`inclk0`) is not toggling, the switchover state machine controls the multiplexer through the `extswitch` signal to switch to the backup clock, `inclk1`.



2.2.10.2. Automatic Switchover with Manual Override

In automatic switchover with manual override mode, you can use the `extswitch` signal for user- or system-controlled switch conditions. You can use this mode for same-frequency switchover, or to switch between inputs of different frequencies.

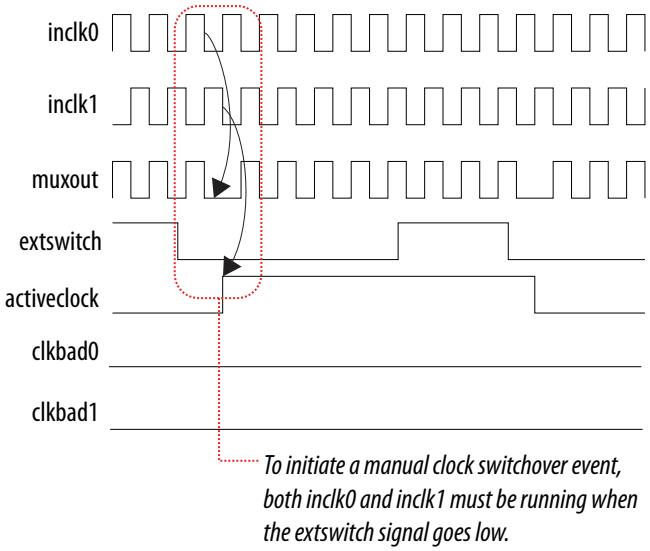
For example, if inclk0 is 66 MHz and inclk1 is 200 MHz, you must control switchover using the extswitch signal. The automatic clock-sense circuitry cannot monitor clock input (inclk0 and inclk1) frequencies with a frequency difference of more than 100% (2×).

This feature is useful when the clock sources originate from multiple cards on the backplane, requiring a system-controlled switchover between the frequencies of operation.

You must choose the backup clock frequency and set the M, N, C, L, and K counters so that the VCO operates within the recommended operating frequency range. The Intel Quartus Prime software notifies you if a given combination of inclk0 and inclk1 frequencies cannot meet this requirement.

Figure 20. Clock Switchover Using the extswitch (Manual) Control

This figure shows a clock switchover waveform controlled by the extswitch signal. In this case, both clock sources are functional and inclk0 is selected as the reference clock. The switchover sequence starts when the extswitch signal goes low. On the falling edge of inclk0, the counter's reference clock, muxout, is gated off to prevent clock glitching. On the falling edge of inclk1, the reference clock multiplexer switches from inclk0 to inclk1 as the I/O PLL reference. The activeclock signal changes to indicate the clock which is currently feeding the I/O PLL.



In automatic override with manual switchover mode, the activeclock signal inverts after the extswitch signal transitions from logic high to logic low. Since both clocks are still functional during the manual switch, neither clkbad signal goes high. Because the switchover circuit is negative-edge sensitive, the rising edge of the extswitch signal does not cause the circuit to switch back from inclk1 to inclk0. When the extswitch signal goes low again, the process repeats.

The extswitch signal and automatic switch work only if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

2.2.10.3. Manual Clock Switchover

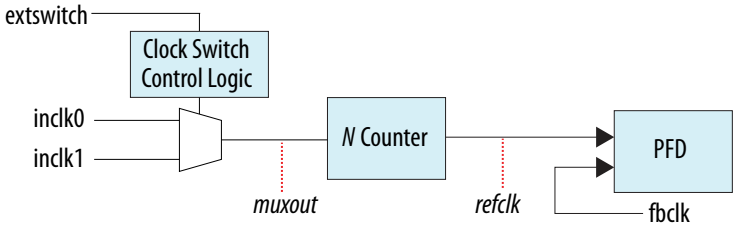
In manual clock switchover mode, the `extswitch` signal controls whether `inclk0` or `inclk1` is selected as the input clock to the I/O PLL. By default, `inclk0` is selected.

A clock switchover event is initiated when the `extswitch` signal transitions from logic high to logic low, and is held low for at least three `inclk` cycles for the `inclk` clock being switched to.

You must bring the `extswitch` signal back high again to perform another switchover event. If you do not require another switchover event, you can leave the `extswitch` signal in a logic low state after the initial switch.

If `inclk0` and `inclk1` are different frequencies and are always running, the `extswitch` signal minimum low time must be greater than or equal to three of the slower frequency `inclk0` and `inclk1` cycles.

Figure 21. Manual Clock Switchover Circuitry in Intel Stratix 10 I/O PLLs



You can delay the clock switchover action by specifying the switchover delay in the Intel FPGA IP cores for the I/O PLL. When you specify the switchover delay, the `extswitch` signal must be held low for at least three `inclk` cycles for the `inclk` being switched to plus the number of the delay cycles that has been specified to initiate a clock switchover.

2.2.11. PLL Reconfiguration and Dynamic Phase Shift

Intel Stratix 10 devices support PLL reconfiguration and dynamic phase shift with the following features:

- PLL reconfiguration—I/O PLL and fPLL are able to reconfigure the `M`, `N`, and `C` counters. Only the fPLL supports reconfiguration of the fractional settings.
- Dynamic phase shift—Only the I/O PLL can perform positive or negative phase shift. Able to shift multiple phase steps each time, where one phase step is equal to 1/8 of the VCO period.

Related Information

- IOPLL Reconfig Intel FPGA IP Core References on page 56
- Reconfiguration Interface and Dynamic Reconfiguration chapter, Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Provides more information about the fPLL reconfiguration.

2.2.12. PLL Calibration

I/O PLLs include both analog and digital blocks that require calibration to compensate for process, voltage, and temperature (PVT) variations. Intel Stratix 10 uses the I/O manager to perform calibration routines.

There are two main types of calibration.

- Power-up calibration—Initiates automatically at device power-up and runs during device configuration.
- User calibration—If you perform dynamic reconfiguration or change the reference clock frequency of the I/O PLL, you must perform user recalibration. You must enable the required calibration sequence.

To successfully complete the calibration process, `OSC_CLK_1` clocks and all reference clocks driving the I/O PLLs must be stable and free running at the start of FPGA configuration. If clock switchover is enabled, both reference clocks must be present for calibration. During user mode, when the I/O PLL does not detect a reference clock during configuration, calibration attempts continue periodically. After calibration has completed, the I/O PLL is locked automatically.

Related Information

Calibration chapter, Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
Provides more information about the fPLL calibration.

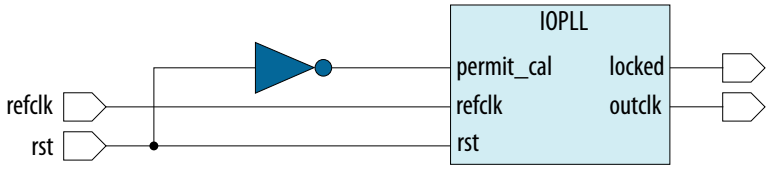
2.2.12.1. Power-Up Calibration

After device power-up, the I/O manager automatically initiates the calibration process. The process continues during device programming.

You must enable the `permit_cal` signal from the IOPLL IP core to delay the power-up calibration in I/O PLL if the reference clock is not stable before device configuration. Set `permit_cal` = 0 upon power up until the reference clock is stable and operating at the correct frequency. Then, set `permit_cal` = 1 to initiate the power-up calibration. You must ensure that the `permit_cal` signal remains high once asserted.

Figure 22. Example of Power-Up Calibration When PLL Reference Clock is Not Stable Upon Power Up

This is an example on how to set `permit_cal` = 0 when the PLL reference clock is not stable before device configuration. You can invert the I/O PLL `reset` signal and connect it to the `permit_cal` port as shown in this figure. Asserting the `reset` signal high delays power-up calibration. Deassert the `reset` signal once the clock driving the I/O PLL is stable to initiate the power-up calibration.



2.2.12.2. User Calibration

The I/O PLL must be recalibrated for any of the following conditions after device power up:

- Dynamic I/O PLL reconfiguration that changes the M or N counter settings is performed.
- Change of the reference clock frequency to the I/O PLL.

Recalibration is not necessary when using clock switchover to a secondary reference clock with a different frequency than the primary reference clock. The I/O PLL stores the calibration settings for both reference clocks after power-up calibration.

To perform the recalibration of the I/O PLL, you must enable the IOPLL Reconfig Intel FPGA IP core to enable the recalibration mode.

Related Information

- Recalibration Using .mif on page 41
- Calibration chapter, Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Provides more information about the fPLL calibration.

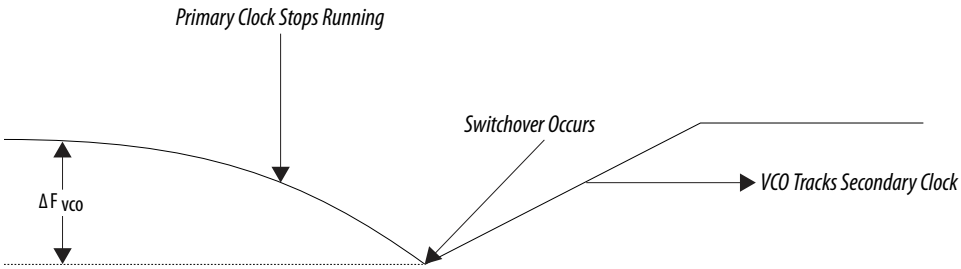
3. Intel Stratix 10 Clocking and PLL Design Considerations

3.1. Guideline: Clock Switchover

When implementing clock switchover in Intel Stratix 10 I/O PLLs, refer to the following guidelines:

- Automatic clock switchover requires that the `inclk0` and `inclk1` frequencies be within 20% of each other. Failing to meet this requirement causes the `clkbad0` and `clkbad1` signals to not function properly.
- When using manual clock switchover, the difference between `inclk0` and `inclk1` can be more than 100% (2×). However, differences in frequency, phase, or both, of the two clock sources may cause the I/O PLL to lose lock. Resetting the I/O PLL ensures that you maintain the correct phase relationships between the input and output clocks.
- Both `inclk0` and `inclk1` must be running when the `extswitch` signal goes low to initiate the manual clock switchover event. Failing to meet this requirement causes the clock switchover to not function properly.
- Applications that require a clock switchover feature and a small frequency drift must use a low-bandwidth I/O PLL. When referencing input clock changes, the low-bandwidth I/O PLL reacts more slowly than a high-bandwidth I/O PLL. When switchover happens, a low-bandwidth I/O PLL propagates the stopping of the clock to the output more slowly than a high-bandwidth I/O PLL. However, be aware that the low-bandwidth I/O PLL also increases lock time.
- After a switchover occurs, there may be a finite resynchronization period for the I/O PLL to lock onto a new clock. The time it takes for the I/O PLL to relock depends on the I/O PLL configuration.
- If the phase relationship between the input clock to the I/O PLL and the output clock from the I/O PLL is important in your design, assert the reset signal for at least 10 ns after performing a clock switchover. Wait for the locked signal to go high and be stable before re-enabling the output clocks from the I/O PLL.
- The VCO frequency gradually decreases when the current clock is lost and then increases as the VCO locks on to the backup clock, as shown in the following figure.

Figure 23. VCO Switchover Operating Frequency



3.2. IP Core Constraints

To implement the fPLL IP core, you must adhere to the following constraints:

- You must use `create_clock` constraints on fPLL reference clocks on the project's top-level SDC file.
- Any SDC design constraints referring to transceiver clocks must be listed after the transceiver Native PHY SDC file constraints.
- fPLL output clocks have no phase relationship to the reference clock when utilizing the fPLL output clocks for core usage. The fPLL output clocks of the clock divider are still in phase with each other, however.

To implement the IOPLL IP core, you must adhere to the following constraints:

- Any SDC design constraints referring to the I/O PLL clocks must be listed after the SDC constraints for the IOPLL IP core.

3.3. Guideline: Resetting the PLL

To reset the PLL, refer to the following guidelines:

- When changing the M counter, N counter, or loop filter settings, the I/O PLL may lose and regain lock. To maintain the appropriate phase relationship between the reference clock and output clocks, assert the `areset` signal to reset the I/O PLL after reconfiguration is complete. Intel recommends always resetting the I/O PLL after any reconfiguration operation to the M counter, N counter, or loop filter settings.
- When changing the C counter settings, you may lose the expected phase relationship between the C counters. Assert the `areset` signal after reconfiguration is complete to restore the expected phase relationship. Reset is not required if the phase relationships are not important to your application.
- Resetting the I/O PLL does not modify the counter or loop filter settings. However, resetting the I/O PLL undoes any dynamic phase shift operations that were performed. After the I/O PLL is reset, the phase shift on the C counters is restored to the originally programmed settings.

3.4. Guideline: Configuration Constraints

The I/O PLL configuration must obey the following constraints:

- The phase frequency detector (PFD) and VCO each have a legal frequency range of operation.
- The loop filter settings must be appropriate for the `M` counter value and user-selected bandwidth mode.

If any of these configuration constraints are violated, the I/O PLL may fail to lock or may exhibit poor jitter performance.

3.5. Guideline: Timing Closure

For timing closure, refer to the following guidelines:

- Reconfiguring a PLL's counter and loop filter settings changes both the output frequency and the clock uncertainty of that I/O PLL. Dynamic phase shift only affects the output clock phase.
- The Timing Analyzer in the Intel Quartus Prime software performs timing analysis for the initial PLL settings only. You must verify that your design closes timing after dynamic reconfiguration or dynamic phase shift.
- Intel recommends compiling the I/O PLL designs with each intended configuration setting to determine the variation in the clock with the I/O PLL settings.

3.6. Guideline: I/O PLL Reconfiguration

To reconfigure the I/O PLL, refer to the following guidelines:

- If the reference clock frequency changes, you must recalibrate the I/O PLL using the IOPLL Intel FPGA IP core.
- The I/O PLL reconfiguration interface must have a free running `mgmt_clk` signal. The I/O PLL dynamic phase shift interface must have a free running `scanclk` signal. These interfaces eliminate the need to precisely control the start and stop of `mgmt_clk` and `scanclk` signals.
- The I/O PLL can be reconfigured with `.mif` streaming mode and advanced mode using the IOPLL Reconfig Intel FPGA IP core. Intel recommends using the `.mif` streaming mode.
- Use caution when reconfiguring an I/O PLL with a non-zero phase shift setting. Modifying the `M` counter or `N` counter settings does not change the relative phase shift (in percent), but alters the absolute phase shift (in picoseconds). Modifying the `C` counter settings does not change the absolute phase shift, but modifies the relative phase shift.

3.7. Guideline: I/O PLL Jitter Performance

To achieve the Intel Stratix 10 I/O PLL clock output jitter performance as specified in the *Intel Stratix 10 Device Datasheet*, adhere to the maximum allowable number of unterminated simultaneously switching output (SSO) pins, such as LVTTL and LVCMOS, within the I/O bank (where the PLL output clock resides).

Table 4. Maximum Allowable Number of Unterminated SSO Pins Within an I/O Bank

SSO Pin Current Strength (mA)	Maximum Allowable Number of SSO Pins
16	17
12	21
10	27
8	36

If you use more than the maximum allowable unterminated SSO pins, additional jitter beyond the PLL clock output jitter specification is induced to the PLL clock output.

Table 5. Jitter Increment per SSO Pin When the Unterminated SSO Pin Utilization of the Affected I/O Bank Exceeds the Maximum Allowable Number

SSO Pin Current Strength (mA)	Jitter Increment per SSO Pin (ps/pin)
16	8
12	7
10	6
8	4

When you use a combination of different current strength, calculate the maximum allowable number of SSO pins according to the strongest current strength used. The jitter increment value per SSO pin should then be calculated using the strongest current strength setting from the remaining SSO pins (after deducting the maximum allowable number of SSO pins from the total SSO pins). For example, if the design has 19 pins with current strength 16 mA, 5 pins with current strength 12 mA, and 2 pins with current strength 10 mA, the maximum allowable number of SSO pins for this design is 17, and the jitter increment per SSO pin is 8 ps/pin. Total additional jitter induced is equivalent to $[(19 + 5 + 2) - 17] \times 8 = 72$ ps.

Related Information

Intel Stratix 10 Device Datasheet
Provides the I/O PLL clock output jitter performance.

3.8. Guideline: Clock Gating

Both sector clock gate and root clock gate have similar skews. To achieve higher performance when transferring between gated and non-gated clocks and vice-versa, refer to the following guidelines:

- Minimize the size of clock region because smaller clock delays have lower skew.
- Ensure that both gated and non-gated clocks have similar sizes.
- Minimize logic on paths between gated and non-gated clocks to achieve better timing margin to compensate for the higher skews.

4. Intel Stratix 10 Clocking and PLL Implementation Guides

If you are looking for the fPLL IP core, refer to the *Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide*.

Related Information

Instantiating the fPLL IP Core, Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide

Provides more information about the fPLL IP core.

4.1. Clock Control Intel Stratix 10 FPGA IP Core

The Clock Control IP core provides clock control features such as enabling entry to the clock network, clock multiplexing, clock gating, and clock division for the Intel Stratix 10 devices.

4.2. IOPLL Intel FPGA IP Core

The IOPLL IP core allows you to configure the settings of the Intel Stratix 10 I/O PLL.

The IOPLL IP core supports the following features:

- Supports six different clock feedback modes: direct, external feedback, normal, source synchronous, zero delay buffer, and LVDS mode.
- Generates up to nine clock output signals for the Intel Stratix 10 device.
- Switches between two reference input clocks.
- Supports adjacent PLL (`adjpllin`) input to connect with an upstream PLL in PLL cascading mode.
- Generates the Memory Initialization File (**.mif**) and allows PLL dynamic reconfiguration.
- Supports PLL dynamic phase shift.

4.2.1. .mif File Generation

You can generate the `.mif` files in the IOPLL IP core parameter editor.

4.2.1.1. Generating a New .mif File

To generate a new `.mif` file containing a single I/O PLL configuration, follow these steps:

1. At the **Dynamic Reconfiguration** tab, select **Enable dynamic reconfiguration of PLL**.
2. For **MIF Generation Options**, select **Generate New MIF File**.
3. For **Path of New MIF file**, specify a file name.
4. For **Name of Current Configuration**, specify the name of the current configuration of the I/O PLL.
5. Click **Create MIF File**.

Related Information

.mif Streaming Reconfiguration on page 40

4.2.1.2. Adding Configurations to Existing .mif File

You can append new configurations to an existing `.mif` file. To store more configurations in a `.mif` file, follow these steps:

1. At the **Dynamic Reconfiguration** tab, select **Enable dynamic reconfiguration of PLL**.
2. For **MIF Generation Options**, select **Add Configuration to Existing MIF File**.
3. For **Path of New MIF file**, specify a file name.
4. For **Name of Current Configuration**, specify the name of the new configuration of the I/O PLL.
5. Click **Append to MIF File**.

Related Information

.mif Streaming Reconfiguration on page 40

4.2.2. Implementing I/O PLL Dynamic Phase Shift in the IOPLL IP Core

You can use the IOPLL IP core to perform phase shifting directly through the dynamic phase shift ports.

4.2.2.1. I/O PLL Dynamic Phase Shift Operation

To perform dynamic phase shift operation for an I/O PLL in the IOPLL IP core, follow these steps:

1. Set the value for `updn`, `cntsel[4..0]`, and `num_phase_shift[2..0]` ports.
2. Assert `phase_en` port for at least two `scanclk` cycles.

Each `phase_en` pulse indicates one dynamic phase shift operation. The `phase_done` output goes low to indicate that dynamic phase shift is in progress. You can only assert the `phase_en` signal after the `phase_done` signal goes from low to high.

The `updn`, `cntsel[4..0]`, and `num_phase_shift[2..0]` ports are synchronous to the `scanclk` cycle.

When the phase_done signal transitions from high to low, the phase_done signal is synchronous to the rising edge of the scanclk signal. The transition from low to high is asynchronous to the scanclk signal.

Depending on the VCO and scanclk frequency, the low time of the phase_done signal may be greater than or less than one scanclk cycle.

Related Information

Dynamic Phase Shift Ports in the IOPLL IP Core on page 54

4.2.3. Design Example

You must install the Intel Quartus Prime software version 17.1 or later. The software must be installed on a Windows* or Linux* computer that meets the Intel Quartus Prime software minimum requirements.

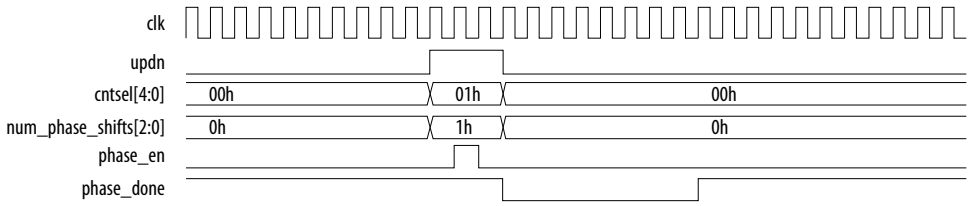
4.2.3.1. Design Example: Dynamic Phase Shift Using IOPLL IP Core

This design example uses the same design as "Design Example 3: Dynamic Phase Shift Using IOPLL Reconfig IP Core" without using the IOPLL Reconfig Intel FPGA IP core. This design example demonstrates the implementation of the I/O PLL dynamic phase shift directly through the IOPLL IP core.

To run the test with this design example, perform these steps:

1. Download and restore the iopll-dynamic-phase-shift.qar file.
2. Change the device and pin assignments of the design example to match your hardware.
3. Recompile the design example. Ensure that the design example does not contain any timing violation after recompilation.
4. Open the AN.stp file and program the device with top.sof.
5. Assert a high pulse on reset_SM signal to start the I/O PLL dynamic phase shift reconfiguration operation.

Figure 24. Waveform Example for Dynamic Phase Shift Using IOPLL IP Core Design Example



Related Information

Design Example: Dynamic Phase Shift Using IOPLL IP Core
Provides the design file for this design example.

4.3. IOPLL Reconfig Intel Stratix 10 FPGA IP Core

You can use Intel Stratix 10 devices to implement phase-locked loop (PLL) reconfiguration and dynamic phase shift for I/O PLLs.

The Intel Stratix 10 I/O PLL supports dynamic reconfiguration when the device is in user mode. With the dynamic reconfiguration feature, you can reconfigure the I/O PLL settings in real time. You can change the divide settings of the PLL counters and the PLL bandwidth settings (loop filter setting and charge pump setting) through an Avalon® Memory-Mapped (Avalon-MM) interface in the IOPLL Reconfig IP core, without the need to reconfigure the entire FPGA. The Intel Stratix 10 I/O PLL uses divide counters (`N`, `M`, and `C` counters) and a voltage-controlled oscillator (VCO) to synthesize the desired phase and frequency output.

You can use the IOPLL Reconfig IP core as follows:

- Memory Initialization File (`.mif`) streaming reconfiguration
  - Allows the I/O PLL reconfiguration using predefined settings saved in an on-chip ROM. You can store many unique PLL configurations in a single ROM.
  - The `.mif` file is generated automatically by the IOPLL IP core. Using the generated `.mif` file during `.mif` streaming reconfiguration ensures the legality of the new configuration.
  - Intel recommends using this reconfiguration method.
- Advanced mode reconfiguration
  - This method of reconfiguration is for advanced users. You must ensure the reconfigured PLL settings are within the legal range.
  - Enable the **Advanced Reconfiguration** option from the IOPLL Reconfig IP core to reconfigure the individual I/O PLL registers.
  - This method is error prone and may lead to the I/O PLL being reconfigured into an illegal configuration if the reconfiguration is done incorrectly.
- Recalibration of the I/O PLL using `.mif`
  - Perform recalibration of the I/O PLL without any reconfiguration.
  - Trigger recalibration if the reference clock frequency changes.
- I/O PLL clock gating
  - Gate and un-gate I/O PLL output clock 0 to output clock 7 of the I/O PLL.

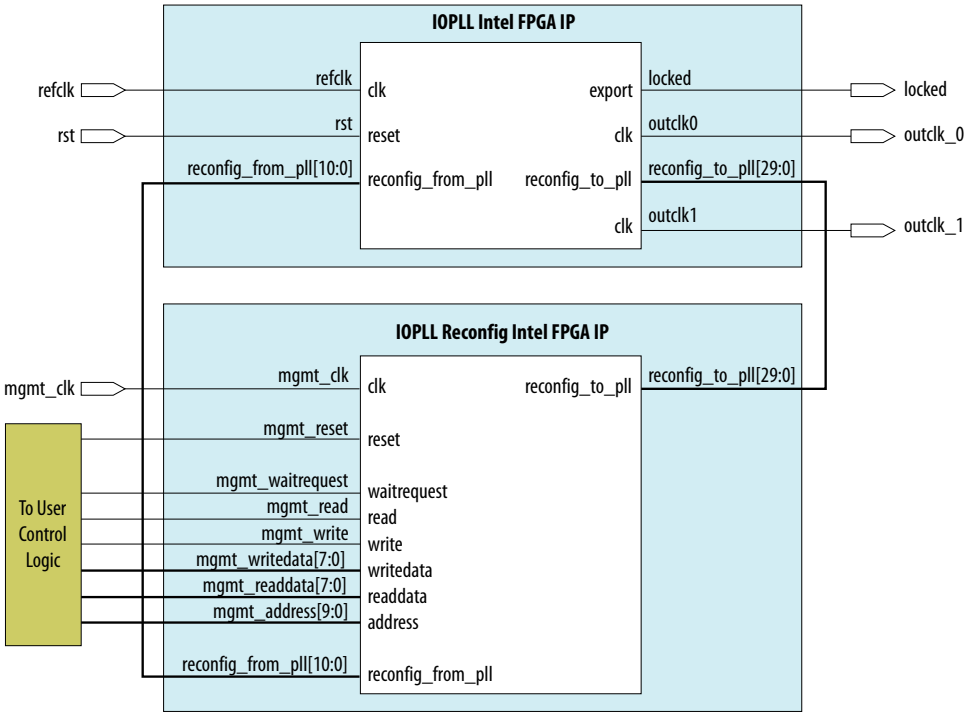
You can perform dynamic phase shift using the IOPLL Reconfig IP core.

4.3.1. Implementing I/O PLL Reconfiguration in the IOPLL Reconfig IP Core

You can enable the PLL reconfiguration circuitry for the I/O PLL through the Avalon-MM interface in the IOPLL Reconfig IP core.

4.3.1.1. Connectivity between the IOPLL and IOPLL Reconfig IP Cores

Figure 25. Connectivity between the IOPLL and IOPLL Reconfig IP Cores in the Intel Quartus Prime Software



Related Information

Avalon-MM Interface Ports in the IOPLL Reconfig IP Core on page 56

4.3.1.2. Connecting the IOPLL and IOPLL Reconfig IP Cores

To connect the IOPLL and IOPLL Reconfig IP cores in your design, follow these steps:

1. Connect the `reconfig_to_pll[29..0]` bus on the IOPLL Reconfig IP core to the `reconfig_to_pll[29..0]` bus on the IOPLL IP core.
2. Connect the `reconfig_from_pll[10..0]` bus on the IOPLL Reconfig IP core to the `reconfig_from_pll[10..0]` bus on the IOPLL IP core.
3. Connect the `mgmt_clk` port to a valid clock source.
4. Connect the `mgmt_reset` port, `mgmt_waitrequest` port, `mgmt_read` port, `mgmt_write` port, `mgmt_readdata[7..0]` bus, `mgmt_writedata[7..0]` bus, and `mgmt_address[9..0]` bus to user control logic to perform read and write operations.

Related Information

Avalon-MM Interface Ports in the IOPLL Reconfig IP Core on page 56

4.3.2. IOPLL Reconfig IP Core Reconfiguration Modes

The IOPLL Reconfig IP core has four functional reconfiguration modes. The reconfiguration operation mode is based on the setting in the `mgmt_address[9:8]` bit.

Table 6. IOPLL Reconfig IP Core Reconfiguration Modes

Reconfiguration Mode	mgmt_address[9:8]
`.mif` streaming reconfiguration	2'b 00
Advanced mode reconfiguration	2'b 01
Clock gating reconfiguration	2'b 10
Dynamic phase shift reconfiguration	2'b 11

After performing dynamic reconfiguration on the I/O PLL that changes the `M` counter, `N` counter, bandwidth setting, or charge pump current, the I/O PLL must be recalibrated. For `.mif` streaming reconfiguration, the recalibration is done automatically. For advanced mode reconfiguration, you must manually trigger the recalibration of the I/O PLL. Recalibration is not needed for clock gating and dynamic phase shift reconfiguration.

4.3.2.1. .mif Streaming Reconfiguration

`.mif` streaming allows you to dynamically reconfigure the I/O PLL through the IOPLL Reconfig IP core using predefined settings saved in an on-chip RAM. You must generate a `.mif` file containing these pre-defined configurations, up to 32 I/O PLL configurations, from the IOPLL IP core parameter editor.

To perform `.mif` streaming reconfiguration, follow these steps:

1. Set `mgmt_address[9:8]` = 2'b00 to choose the `.mif` streaming mode and set `mgmt_writedata[4:0]` to the index of the desired configuration in the `.mif` file.
2. To start the `.mif` streaming reconfiguration on the I/O PLL, assert the `mgmt_write` signal for one `mgmt_clk` cycle. `mgmt_waitrequest` is asserted by the IOPLL Reconfig IP core while `.mif` streaming is in progress.
3. After the reconfiguration is complete, the `mgmt_waitrequest` signal is de-asserted.
4. In the IOPLL Reconfig IP core parameter editor, select the **Assert waitrequest until IOPLL has locked** option for the I/O PLL to lock. Otherwise, you can wait for the I/O PLL to lock to ensure the I/O PLL reconfiguration is complete.

Recalibration is done automatically for `.mif` streaming reconfiguration. If you want to manually trigger recalibration using `.mif` file, follow the steps in the *Recalibration Using .mif* section.

Related Information

- Generating a New .mif File on page 36
- Adding Configurations to Existing .mif File on page 36
- Recalibration Using .mif on page 41

4.3.2.1.1. Recalibration Using .mif

Recalibration using `.mif` only allows you to recalibrate the I/O PLL but not to reconfigure the I/O PLL. In the IOPLL Reconfig IP core, enable **Recalibration Mode**. When the recalibration is selected, a `recalibration.mif` file is generated automatically for the recalibration operation.

To perform I/O PLL recalibration using `.mif`, follow these steps:

1. Set `mgmt_address[9:8]` = 2'b00 to choose the `.mif` mode and set `mgmt_writedata[4:0]` = 2'b00.
2. To start the recalibration using `.mif` on the I/O PLL, assert the `mgmt_write` signal for one `mgmt_clk` cycle. `mgmt_waitrequest` is asserted by the IOPLL Reconfig IP core while recalibration is in progress.
3. After the recalibration is complete, the `mgmt_waitrequest` signal is deasserted.

4.3.2.2. Advanced Mode Reconfiguration

In advanced mode, individual I/O PLL setting is reconfigured using the IOPLL Reconfig IP core through the Avalon interface.

Advanced mode reconfiguration is only recommended for advanced users. This reconfiguration mode has several limitations and can cause the I/O PLL to lose lock and can lead to device reliability problems if you set the configuration parameters to the illegal configuration settings. Intel recommends using the `.mif` streaming reconfiguration.

The limitations of using the advanced mode reconfiguration are as follows:

- You must ensure that the configuration setting is a legal value so that the I/O PLL has a legal configuration. To ensure your configuration is legal, refer to the *IOPLL IP Core Parameters - Advanced Parameters Tab* table for the correct configuration settings.
- If the value to be reconfigured makes up only a part of one byte in the I/O PLL's internal memory, you must perform a read-modify-write operation to not overwrite the remaining bits of the byte.
- You must manually trigger recalibration of the I/O PLL after the advanced mode reconfiguration.

Caution: PLL may lose lock and can cause reliability problems to your device if you configure with the wrong PLL setting, configure the wrong bit, or overwrite the whole byte for settings that made up just part of one byte.

To perform I/O PLL reconfiguration using advanced mode, follow these steps:

1. Enable the **Advanced Reconfiguration** option in the IOPLL Reconfig IP core.
2. Set `mgmt_address[9:8]` = 2'b01 to choose the advanced mode reconfiguration.
3. Set the address bus value for `mgmt_address[7:0]` and the data bus value for `mgmt_writedata [7:0]` as the desired PLL setting.

   For more details, refer to the *Address Bus and Data Bus Settings for Advanced Mode Reconfiguration* table.

4. Assert the `mgmt_write` signal for one `mgmt_clk` cycle.
5. Repeat step 1 until step 3 to set address bus and data bus value for the desired I/O PLL reconfiguration setting.
6. After the I/O PLL reconfiguration is complete, you must manually trigger the I/O PLL recalibration.

   For more details about the I/O PLL recalibration, refer to the *Recalibration Using Advanced Mode* section.

Related Information

- .mif Streaming Reconfiguration on page 40
- IOPLL IP Core Parameters - Advanced Parameters Tab on page 53
- Address Bus and Data Bus Settings for Advanced Mode Reconfiguration on page 56
- Recalibration Using Advanced Mode on page 42

4.3.2.2.1. Recalibration Using Advanced Mode

To perform I/O PLL recalibration using advanced mode, follow these steps:

1. Set `mgmt_address[9:8]` = 2'b01 to choose the advanced mode.
2. Set the `mgmt_writedata[6]` to 1'b1 on `mgmt_address[7:0]` = 8'b01001001 by performing the read-modify-write operation.
3. Set `mgmt_address[7:0]` = 8'b01001010 and `mgmt_wriredata[7:0]` = 8'b00000011 to enable the calibration interface.

4.3.2.3. Clock Gating Reconfiguration

You can gate (disable) and un-gate (enable) I/O PLL output clock 0 to output clock 7 of the I/O PLL. It is easily done by writing one byte to the IOPLL Reconfig IP core, with one bit corresponding to each of the I/O PLL output clocks.

To perform clock gating reconfiguration, follow these steps:

1. Set `mgmt_address[9:8]` to 2'b10 to select clock gating mode and set `mgmt_writedata[7:0]` to indicate desired output clock to be gated.
2. To start the clock gating reconfiguration on the I/O PLL, assert the `mgmt_write` signal for one `mgmt_clk` cycle.
3. The gating changes may not come into effect for multiple clock cycles after `mgmt_waitrequest` has been de-asserted.

Related Information

Output Clock and the Corresponding Data Bit Setting for Clock Gating Reconfiguration on page 62

4.3.2.4. Dynamic Phase Shift Reconfiguration

The dynamic phase shifts reconfiguration can determine the number of shifts, the direction of the phase shift and the output clock to be shifted.

To perform dynamic phase shift reconfiguration through the IOPLL Reconfig IP core, follow these steps:

1. Set `mgmt_address[9:8]` to 2'b11 to select dynamic phase shift reconfiguration mode.
2. set `mgmt_writedata[7:0]` to indicate the desired number of phase shift, the direction of phase shift, and the desired counter to be shifted.
3. To start the dynamic phase shift reconfiguration on the I/O PLL, assert the `mgmt_write` signal for one `mgmt_clk` cycle. This signal is the equivalent of the `phase_en` signal on the I/O PLL.
4. After the dynamic phase shift is complete, the `mgmt_waitrequest` signal is de-asserted.

Related Information

Data Bus Setting for Dynamic Phase Shift for IOPLL Reconfig IP Core on page 62

4.3.3. Design Examples

You must install the Intel Quartus Prime software version 17.1 or later. The software must be installed on a Windows or Linux computer that meets the Intel Quartus Prime software minimum requirements.

4.3.3.1. Design Example 1: .mif Streaming Reconfiguration Using IOPLL Reconfig IP Core

This design example uses a 1SG280LU3F50E2VGS1 device to demonstrate the implementation of the I/O PLL reconfiguration through `.mif` streaming using the IOPLL Reconfig IP core. This design example consists of the IOPLL IP core, IOPLL Reconfig IP core, and In-System Sources & Probes Intel FPGA IP core.

The I/O PLL synthesizes two output clocks of 400 MHz with 0 ps phase shift and 200 MHz with 0 ps phase shift on counter `C0` output and counter `C1` output respectively at medium bandwidth. The input reference clock is 50 MHz.

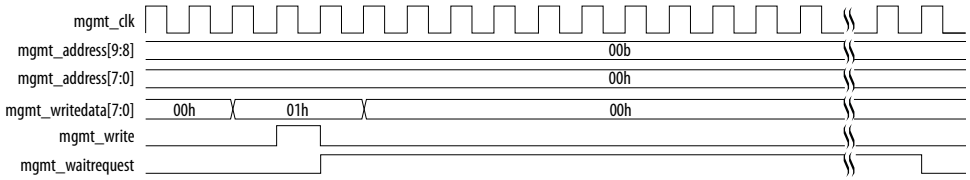
The IOPLL Reconfig IP core connects to a state machine to perform the I/O PLL `.mif` streaming reconfiguration operation. A high pulse on the `reset_SM` input through the In-System Sources & Probes IP core triggers the I/O PLL reconfiguration operation. After the I/O PLL reconfiguration operation is complete, the I/O PLL operates in the following configuration at medium bandwidth:

- 100 MHz with 0 ps phase shift on counter `C0` output
- 100 MHz with 0 ps phase shift on counter `C1` output

To run the test with this design example, perform these steps:

1. Download and restore the `iopll-reconfig-mif-streaming.qar` file.
2. Change the device and pin assignments of the design example to match your hardware.
3. Recompile the design example. Ensure that the design example does not contain any timing violation after recompilation.
4. Open the `AN.stp` file and program the device with `top.sof`.
5. Assert a high pulse on the `reset_SM` signal to start the I/O PLL reconfiguration operation.

Figure 26. Waveform Example for .mif Streaming Reconfiguration Design Example



Related Information

Design Example 1: .mif Streaming Reconfiguration Using IOPLL Reconfig IP Core
Provides the design file for this design example.

4.3.3.2. Design Example 2: Advanced Mode Reconfiguration Using IOPLL Reconfig IP Core

This design example uses a 1SX280LU2f50E2VGS2 device to demonstrate the implementation of the I/O PLL reconfiguration in advanced mode using the IOPLL Reconfig IP core. This design example consists of the IOPLL IP core, IOPLL Reconfig IP core, and In-System Sources & Probes IP core.

The I/O PLL synthesizes two output clocks of 400 MHz and 200 MHz on counter `C0` output and counter `C1` output respectively at medium bandwidth. The input reference clock is 50 MHz.

The IOPLL Reconfig IP core connects to a state machine to perform I/O PLL reconfiguration operation. A high pulse on the `reset_SM` input through In-System Sources & Probes IP core triggers the I/O PLL reconfiguration operation. After the I/O PLL reconfiguration operation is complete, the I/O PLL operates in the following configuration at medium bandwidth:

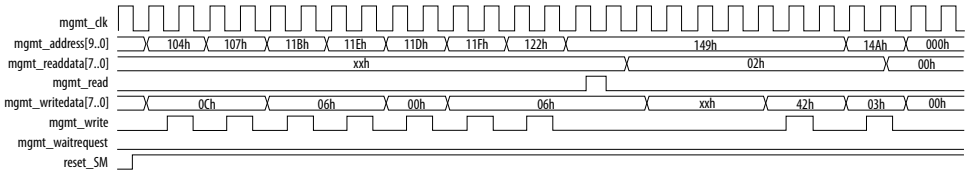
- 100 MHz on counter `C0` output
- 100 MHz on counter `C1` output

The state machine initiates the I/O PLL recalibration process when the I/O PLL reconfiguration operation is complete.

To run the test with this design example, perform these steps:

1. Download and restore the `iopll-reconfig-advanced-mode.qar` file.
2. Change the device and pin assignments of the design example to match your hardware.
3. Recompile the design example. Ensure that the design example does not contain any timing violation after reconfiguration.
4. Open the `AN.stp` file and program the device with `top.sof`.
5. Assert a high pulse on the `mgmt_reset` signal to reset the IOPLL Reconfig IP core.
6. Assert a high pulse on the `reset_SM` signal to start the I/O PLL reconfiguration operation.

Figure 27. Waveform Example for Advanced Mode Reconfiguration Design Example



Related Information

Design Example 2: Advanced Mode Reconfiguration Using IOPLL Reconfig IP Core
Provides the design file for this design example.

4.3.3.3. Design Example 3: Clock Gating Reconfiguration Using IOPLL Reconfig IP Core

This design example uses a 1SG280LU3F50E2VGS1 device to demonstrate the implementation of the I/O PLL clock gating reconfiguration using the IOPLL Reconfig IP core. This design example consists of the IOPLL IP core, IOPLL Reconfig IP core, and In-System Sources & Probes IP core.

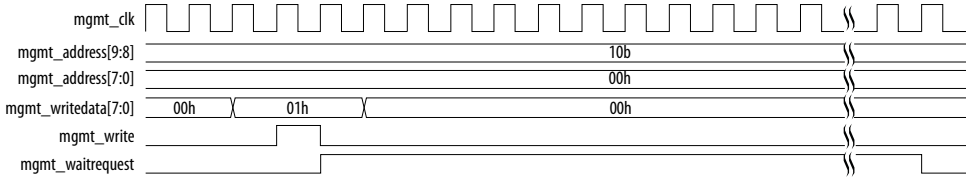
The I/O PLL synthesizes eight output clocks of 200 MHz each. The input reference clock is 50 MHz.

The IOPLL Reconfig IP core connects to a state machine to perform the I/O PLL clock output gating. A high pulse on the `reset_SM` input through the In-System Sources & Probes IP core triggers the I/O PLL reconfiguration operation. After the I/O PLL reconfiguration operation is complete, `outclk0` is ungated and `outclk1` is gated.

To run the test with this design example, perform these steps:

1. Download and restore the `iopll-reconfig-clock-gating.qar` file.
2. Change the device and pin assignments of the design example to match your hardware.
3. Recompile the design example. Ensure that the design example does not contain any timing violation after recompilation.
4. Open the `AN.stp` file and program the device with `top.sof`.
5. Assert a high pulse on the `reset_SM` signal to start the I/O PLL clock gating reconfiguration operation.

Figure 28. Waveform Example for Clock Gating Reconfiguration Design Example



Related Information

Design Example 3: Clock Gating Reconfiguration Using IOPLL Reconfig IP Core
Provides the design file for this design example.

4.3.3.4. Design Example 4: Dynamic Phase Shift Using IOPLL Reconfig IP Core

This design example uses a 1SG280LU3F50E2VGS1 device to demonstrate the implementation of the I/O PLL dynamic phase shift reconfiguration using the IOPLL Reconfig IP core. This design example consists of the IOPLL IP core, IOPLL Reconfig IP core, and In-System Sources & Probes IP core.

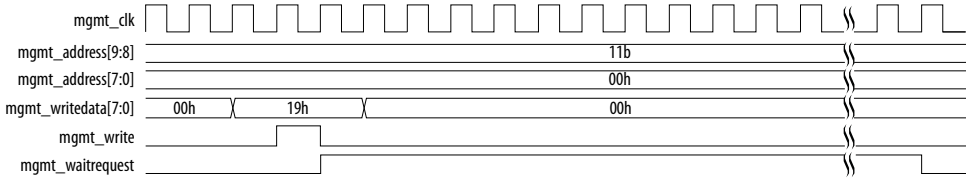
The I/O PLL synthesizes two output clocks of 200 MHz with 0 ps phase shift on counter `C0` output and counter `C1` output at medium bandwidth. The input reference clock is 50 MHz.

The IOPLL Reconfig IP core connect to a state machine to perform the I/O PLL dynamic phase shift operation. A high pulse on the `reset_SM` input through the In-System Sources & Probes IP core triggers the I/O PLL dynamic phase shift operation. After the I/O PLL dynamic phase shift operation is complete, counter `C1` is phase shifted 89 ps for one positive phase shift step.

To run the test with this design example, perform these steps:

1. Download and restore the `iopll-reconfig-dynamic-phase-shift.qar` file.
2. Change the device and pin assignments of the design example to match your hardware.
3. Recompile the design example. Ensure that the design example does not contain any timing violation after recompilation.
4. Open the `AN.stp` file and program the device with `top.sof`.
5. Assert a high pulse on the `reset_SM` signal to start the I/O PLL dynamic phase shift reconfiguration operation.

Figure 29. Waveform Example for Dynamic Phase Shift Using IOPLL Reconfig IP Core Design Example



Related Information

Design Example 4: Dynamic Phase Shift Using IOPLL Reconfig IP Core
Provides the design file for this design example.

5. Clock Control Intel Stratix 10 FPGA IP Core References

5.1. Clock Control IP Core Parameters

Table 7. Clock Control IP Core Parameters for Intel Stratix 10 Devices

Parameter	Legal Value	Description
Number of Clock Inputs	**1**, **2**, or **4**	Specify the number of input clock sources for the clock control block. You can specify up to four clock inputs. Clock multiplexing in Intel Stratix 10 devices is implemented using soft logic in the core.
Ensure glitch free clock switchover	On or Off	Turn on this option to implement a glitch-free switchover when you use multiple clock inputs. You must ensure the currently selected clock is running before switching to another source. If the selected clock is not running, you cannot switch to the new clock source using the glitch-free switchover implementation. By default, the `clkselect` port is set to `00`. You must apply a clock to `inclk0x` to read the values on the `clkselect` ports. This feature will be available in a future release.
Clock Enable	On or Off	Turn on this option if you want to gate your clock output with an enable signal. This option disables the option to use clock division.
Clock Enable Type	**Root Level** or **Distributed Sector Level**	Select the clock gates located in the periphery or the gates located in the sector. For more information about the clock gates, refer to the Clock Gating section.
Enable Register Mode	**Negative Latch** or **None**	Specify if the enable signal should be latched.
Clock Divider	On or Off	Turn on this option if you want to use the clock division block in the periphery.
Clock Divider Output Ports	**Divide 1x**, **Divide 1x and 2x**, or **Divide 1x, 2x and 4x**	Specify the combination of passing your clock through, dividing your clock by 2, or dividing your clock by 4.

Related Information

Clock Gating on page 10

5.2. Clock Control IP Core Ports and Signals

Table 8. Clock Control IP Core Ports for Intel Stratix 10 Devices

Port Name	Description
`inclk`	Input signal to the clock network.
`inclk0x`, `inclk1x`, `inclk2x`, `inclk3x`	Input signals to the clock network based on the value selected for the **Number of Clock Inputs** parameter.

continued...

Port Name	Description
clkselect[]	Input that dynamically selects the clock source to drive the clock network that is driven by the clock buffer. Input port [1 DOWNTO 0] wide. The following list shows the signal selection for the clkselect[] value: • 2'b00 selects inclk0x • 2'b01 selects inclk1x • 2'b10 selects inclk2x • 2'b11 selects inclk3x
outclk	Output of the Clock Control IP core when **Clock Divider** option is not selected.
ena	Clock enable of the clock gate block. This signal is active-high.
clock_div1x, clock_div2x, clock_div4x	Outputs of the Clock Control IP core when the **Clock Divider** option is selected. The exact combination of ports exposed depends on the value specified for the **Clock Divider Output Ports** parameter. • clock_div1x is the same as inclk • clock_div2x divides inclk by 2 • clock_div4x divides inclk by 4

6. IOPLL Intel FPGA IP Core References

6.1. IOPLL IP Core Parameters

The IOPLL IP core parameter editor appears in the PLL category of the IP Catalog.

6.1.1. IOPLL IP Core Parameters - PLL Tab

Table 9. IOPLL IP Core Parameters - PLL Tab for Intel Stratix 10 Devices

Parameter	Legal Value	Description
Device Family	Intel Stratix 10	Specifies the device family.
Component	—	Specifies the targeted device.
Speed Grade	—	Specifies the speed grade for targeted device.
PLL Mode	**Integer-N PLL**	Specifies the mode used for the IOPLL IP core. The only legal selection is **Integer-N PLL**.
Reference Clock Frequency	—	Specifies the input frequency for the input clock, `refclk`, in MHz. The default value is **100.0 MHz**. The minimum and maximum value is dependent on the selected device.
My reference clock frequency might change	Turn on or Turn off	Select this option if you expect the reference clock frequency to change at runtime.
Enable Locked Output Port	Turn on or Turn off	Turn on to enable the `locked` port.
Enable physical output clock parameters	Turn on or Turn off	Turn on to enter physical PLL counter parameters instead of specifying a desired output clock frequency.
Operation Mode	**direct**, **external feedback**, **normal**, **source synchronous**, **zero delay buffer**, or **lvds**	Specifies the operation of the PLL. The default operation is **direct** mode. • If you select the **direct** mode, the PLL minimizes the length of the feedback path to produce the smallest possible jitter at the PLL output. The internal-clock and external-clock outputs of the PLL are phase-shifted with respect to the PLL clock input. In this mode, the PLL does not compensate for any clock networks. • If you select the **external feedback** mode, you must connect the `fbclk` input port to an input pin. A board-level connection must connect both the input pin and external clock output port, `fboutclk`. The `fbclk` port is aligned with the input clock. • If you select the **normal** mode, the PLL compensates for the delay of the internal clock network used by the clock output. If the PLL is also used to drive an external clock output pin, a corresponding phase shift of the signal on the output pin occurs.

continued...

Parameter	Legal Value	Description
		• If you select the **source synchronous** mode, the clock delay from pin to I/O input register matches the data delay from pin to I/O input register. • If you select the **zero delay buffer** mode, the PLL must feed an external clock output pin and compensate for the delay introduced by that pin. The signal observed on the pin is synchronized to the input clock. The PLL clock output connects to the `altbidir` port and drives `zdbfbclk` as an output port. If the PLL also drives the internal clock network, a corresponding phase shift of that network occurs. • If you select the **lvds** mode, the same data and clock timing relationship of the pins at the internal SERDES capture register is maintained. The mode compensates for the delays in LVDS clock network, and between the data pin and clock input pin to the SERDES capture register paths.
Compensated Outclk (5)	0–8	Allows you to select which output clock (`outclk`) to be compensated. The feedback mode compensates for the clock network delay of the `outclk` selected. This feedback mode ensures correct phase relationship between I/O PLL input and output clocks only for the selected `outclk`.
Use Nondedicated Feedback Path (5)	Turn on or Turn off	Turn on to conserve clock resources and improve timing analysis. However, this feature creates frequency limitations and disables phase shift.
Number of Clocks	1–9	Specifies the number of output clocks required for each device in the PLL design. The requested settings for output frequency, phase shift, and duty cycle are shown based on the number of clocks selected.
Multiply Factor (M-Counter) (6)	4–160	Specifies the multiply factor of M-counter.
Divide Factor (N-Counter) (6)	1–110	Specifies the divide factor of N-counter.
Specify VCO Frequency	Turn on or Turn off	Allows you to restrict the VCO frequency to the specified value. This is useful when creating a PLL for LVDS external mode, or if a specific dynamic phase shift step size is desired.
VCO Frequency (7)	—	• When **Enable physical output clock parameters** is turned on—displays the VCO frequency based on the values for **Reference Clock Frequency**, **Multiply Factor (M-Counter)**, and **Divide Factor (N-Counter)**. • When **Enable physical output clock parameters** is turned off—allows you to specify the requested value for the VCO frequency. The default value is **600.0 MHz**.
Give clock global name	Turn on or Turn off	Allows you to rename the output clock name.
Clock Name	—	The user clock name for Synopsis Design Constraints (SDC).
Divide Factor (C-Counter) (6)	1-510	Specifies the divide factor for the output clock (C-counter).

continued...

(5) This option is only available when either **normal** or **source synchronous** mode is selected.

(6) This parameter is only available when **Enable physical output clock parameters** is turned on.

(7) This parameter is only available when **Enable physical output clock parameters** is turned off.

Parameter	Legal Value	Description
Desired Frequency	—	Specifies the output clock frequency of the corresponding output clock port, `outclk[]`, in MHz. The default value is **100.0 MHz**. The minimum and maximum values depend on the device used. The PLL only reads the numerals in the first six decimal places.
Actual Frequency	—	Allows you to select the actual output clock frequency from a list of achievable frequencies. The default value is the closest achievable frequency to the desired frequency.
Phase Shift units	**ps** or **degrees**	Specifies the phase shift unit for the corresponding output clock port, `outclk[]`, in picoseconds (ps) or degrees.
Desired Phase Shift	—	Specifies the requested value for the phase shift. The default value is **0 ps**.
Actual Phase Shift	—	Allows you to select the actual phase shift from a list of achievable phase shift values. The default value is the closest achievable phase shift to the desired phase shift.
Desired Duty Cycle	0.0–100.0	Specifies the requested value for the duty cycle. The default value is **50.0%**.
Actual Duty Cycle	—	Allows you to select the actual duty cycle from a list of achievable duty cycle values. The default value is the closest achievable duty cycle to the desired duty cycle.

6.1.2. IOPLL IP Core Parameters - Settings Tab

Table 10. IOPLL IP Core Parameters - Settings Tab for Intel Stratix 10 Devices

Parameter	Legal Value	Description
PLL Bandwidth Preset	**Low**, **Medium**, or **High**	Specifies the PLL bandwidth preset setting. The default selection is **Low**.
Lock Threshold Setting	**Low Lock Time**, **Medium Lock Time**, or **High Lock Time**	This setting determines the sensitivity of the I/O PLL when detecting lock. This is a tradeoff between the time it takes to lock and the accuracy of the `outclk` frequency when `locked` is first asserted. For applications that require the I/O PLL to lock quickly, **Low Lock Time** is the best option. The estimated lock times are 30 µs + *a* × `refclk_period`, where *a* is 100, 2048, and 4095 for **Low Lock Time**, **Medium Lock Time**, and **High Lock Time** respectively.
PLL Auto Reset	Turn on or Turn off	Automatically self-resets the PLL on loss of lock.
Create a second input clk 'refclk1'	Turn on or Turn off	Turn on to provide a backup clock attached to your PLL that can switch with your original reference clock.
Second Reference Clock Frequency (8)	—	Selects the frequency of the second input clock signal. The default value is **100.0 MHz**. The minimum and maximum value is dependent on the device used.
Create an 'active_clk' signal to indicate the input clock in use (8)	Turn on or Turn off	Turn on to create the `activeclk` output. The `activeclk` output indicates the input clock which is in use by the PLL. Output signal low indicates `refclk` and output signal high indicates `refclk1`.
Create a 'clkbad' signal for each of the input clocks (8)	Turn on or Turn off	Turn on to create two `clkbad` outputs, one for each input clock. Output signal low indicates the clock is working and output signal high indicates the clock is not working.

continued...

(8) This parameter is only available when **Create a second input clk 'refclk1'** is turned on.

Parameter	Legal Value	Description
Switchover Mode (8)	**Automatic Switchover**, **Manual Switchover**, or **Automatic Switchover with Manual Override**	Specifies the switchover mode for design application. The IP supports three switchover modes: • If you select the **Automatic Switchover** mode, the PLL circuitry monitors the selected reference clock. If one clock stops, the circuit automatically switches to the backup clock in a few clock cycles and updates the status signals, `clkbad` and `activeclk`. • If you select the **Manual Switchover** mode, when the control signal, `extswitch`, changes from logic low to logic high, and stays high for at least three clock cycles, the input clock switches to the other clock. The `extswitch` can be generated from FPGA core logic or input pin. • If you select **Automatic Switchover with Manual Override** mode, when the `extswitch` signal is high, it overrides the automatic switch function. As long as `extswitch` remains high, further switchover action is blocked. To select this mode, your two clock sources must be running and the frequency of the two clocks cannot differ by more than 20%. If both clocks are not on the same frequency, but their period difference is within 20%, the clock loss detection block detects the lost clock. The PLL most likely drops out of lock after the PLL clock input switchover and needs time to lock again.
Switchover Delay (8)	0–7	Adds a specific amount of cycle delay to the switchover process.
Access to PLL LVDS_CLK/ LOADEN output port	**Disabled**, **Enable LVDS_CLK/ LOADEN 0**, or **Enable LVDS_CLK/ LOADEN 0 & 1**	Select **Enable LVDS_CLK/LOADEN 0** or **Enable LVDS_CLK/ LOADEN o & 1** to enable the PLL `lvds_clk` or `loaden` output port. Enables this parameter in case the PLL feeds an LVDS SERDES block with external PLL. When using the I/O PLL `outclk` ports with LVDS ports, `outclk[0..3]` are used for `lvds_clk[0,1]` and `loaden[0,1]` ports, `outclk4` can be used for `coreclk` ports.
Enable access to the PLL DPA output port	Turn on or Turn off	Turn on to enable the PLL DPA output port.
Enable access to PLL external clock output port	Turn on or Turn off	Turn on to enable the PLL external clock output port.
Specifies which outclk to be used as extclk_out[0] source	C0 – C8	Specifies the `outclk` port to be used as `extclk_out[0]` source.
Specifies which outclk to be used as extclk_out[1] source	C0 – C8	Specifies the `outclk` port to be used as `extclk_out[1]` source.

6.1.3. IOPLL IP Core Parameters - Cascading Tab

Table 11. IOPLL IP Core Parameters - Cascading Tab

Parameter	Legal Value	Description
Create a 'cascade out' signal to connect with a downstream PLL	Turn on or Turn off	Turn on to create the `cascade_out` port, which indicates that this PLL is a source and connects with a destination (downstream) PLL.
Specifies which outclk to be used as cascading source	0-8	Specifies the cascading source.
Create an adjpllin or cclk signal to connect with an upstream PLL	Turn on or Turn off	Turn on to create an input port, which indicates that this PLL is a destination and connects with a source (upstream) PLL.
Create a permit_cal signal to connect with an upstream PLL	Turn on or Turn off	Turn on to create an input port to enable destination (downstream) PLL power-up calibration. Connect source (upstream) PLL `locked` signal to this input port.

6.1.4. IOPLL IP Core Parameters - Dynamic Reconfiguration Tab

Table 12. IOPLL IP Core Parameters - Dynamic Reconfiguration Tab for Intel Stratix 10 Devices

Parameter	Legal Value	Description
Enable dynamic reconfiguration of PLL	Turn on or Turn off	Turn on to enable the dynamic reconfiguration of this PLL (in conjunction with the IOPLL Reconfig Intel FPGA IP core).
Enable access to dynamic phase shift ports	Turn on or Turn off	Turn on to enable the dynamic phase shift interface with the PLL.
MIF Generation Option (9)	**Generate New MIF File**, **Add Configuration to Existing MIF File**, or **Create MIF File during IP Generation**	Either create a new `.mif` file containing the current configuration of the I/O PLL by clicking **Create MIF File** or add this configuration to an existing `.mif` file by clicking **Append to MIF File**. A `.mif` file also can be opted to be generated during IP generation. The generated `.mif` file contains current PLL profile and a collection of physical parameters—such as `M`, `N`, `C`, `K`, bandwidth, and charge pump—that defines that PLL. You can use this `.mif` file during dynamic reconfiguration to reconfigure the I/O PLL to its current settings.
Path to New/Existing MIF file (9)	—	Enter location and file name of the new `.mif` file to be created or existing `.mif` file to be appended.
Name of Current Configuration (9)	—	Enter the file name of the existing `.mif` file you intend to add to.

6.1.5. IOPLL IP Core Parameters - Advanced Parameters Tab

Table 13. IOPLL IP Core Parameters - Advanced Parameters Tab for Intel Stratix 10 Devices

Parameter	Legal Value	Description
Advanced Parameters	—	Displays a table of physical PLL settings that are implemented based on your input.

6.2. IOPLL IP Core Ports and Signals

Table 14. IOPLL IP Core Ports for Intel Stratix 10 Devices

Port Name	Type	Condition	Description
`refclk`	Input	Required	The reference clock source that drives the I/O PLL.
`rst`	Input	Required	The asynchronous reset port for the output clocks. Drive this port high to reset all output clocks to the value of 0.
`fbclk`	Input	Optional	The external feedback input port for the I/O PLL. The IOPLL IP core creates this port when the I/O PLL is operating in external feedback mode or zero-delay buffer mode. To complete the feedback loop, a board-level connection must connect the `fbclk` port and the external clock output port of the I/O PLL.
`fboutclk`	Output	Optional	The port that feeds the `fbclk` port through the mimic circuitry.

continued...

(9) This parameter is only available when **Enable dynamic reconfiguration of PLL** is turned on.

Port Name	Type	Condition	Description
			The `fboutclk` port is available only if the I/O PLL is in external feedback mode.
`zdbfbclk`	Bidirectional	Optional	The bidirectional port that connects to the mimic circuitry. This port must connect to a bidirectional pin that is placed on the positive feedback dedicated output pin of the I/O PLL. The `zdbfbclk` port is available only if the I/O PLL is in zero-delay buffer mode.
`locked`	Output	Optional	The IOPLL IP core drives this port high when the PLL acquires lock. The port remains high as long as the I/O PLL is locked. The I/O PLL asserts the `locked` port when the phases and frequencies of the reference clock and feedback clock are the same or within the lock circuit tolerance. When the difference between the two clock signals exceeds the lock circuit tolerance, the I/O PLL loses lock.
`refclk1`	Input	Optional	Second reference clock source that drives the I/O PLL for clock switchover feature.
`extswitch`	Input	Optional	Active low signal. Assert the `extswitch` signal low (1'b0) for at least three clock cycles to manually switch the clock.
`activeclk`	Output	Optional	Output signal to indicate which reference clock source is in used by I/O PLL.
`clkbad`	Output	Optional	Output signal that indicates the status of reference clock source is good or bad.
`cascade_out`	Output	Optional	Output signal that feeds into downstream I/O PLL.
`adjpllin`	Input	Optional	Input signal that feeds from upstream I/O PLL.
`outclk_[]`	Output	Optional	Output clock from I/O PLL.
`permit_cal`	Input	Optional	This is an input port for the downstream I/O PLL. Connect this `permit_cal` port to the `locked` output port of the upstream I/O PLL. Connecting this `permit_cal` port ensures that the cascaded I/O PLLs are calibrated in the correct order.

6.3. Dynamic Phase Shift Ports in the IOPLL IP Core

Figure 30. Dynamic Phase Shift Port Ports in the IOPLL IP Core

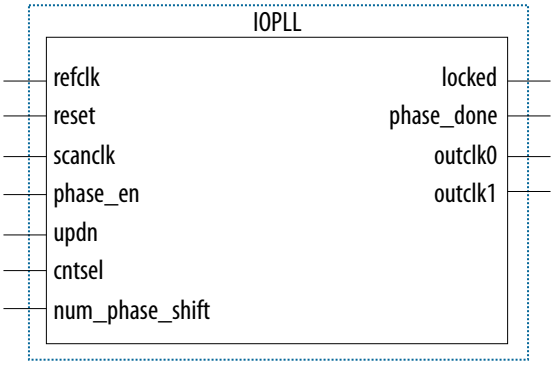


Table 15. Dynamic Phase Shift Ports in the IOPLL IP Core

Port	Direction	Description
scanclk	Input	Dynamic phase shift clock that drives the IOPLL IP core dynamic phase shift operation. This port must be connected to a valid clock source. The maximum input clock frequency is 100 MHz.
phase_en	Input	Active high signal. Asserts to start the dynamic phase shift operation. phase_en can only be asserted 4 clocks after phase_done assertion.
updn	Input	Determines the direction of dynamic phase shift. When updn = 0, phase shift is in negative direction. When updn = 1, phase shift is in positive direction.
cntsel[4..0]	Input	Determines the counter to be selected to perform dynamic phase shift operation.

Counter Name	cntsel[4..0] (Binary)
C0	5'b00000
C1	5'b00001
C2	5'b00010
C3	5'b00011
C4	5'b00100
C5	5'b00101
C6	5'b00110
C7	5'b00111
C8	5'b01000
All C counters	5'b01111

num_phase_shift[2..0]	Input	Determines the number of phase shifts per dynamic phase shift operation. Up to seven phase shifts per operation are possible. Each phase shift step is equal to 1/8 of I/O PLL VCO period. num_phase_shift must never be set to 0 in DPS mode.
phase_done	Output	The IOPLL IP core drives this port high for one scanclk cycle after dynamic phase shift operation is complete.

7. IOPLL Reconfig Intel FPGA IP Core References

7.1. Avalon-MM Interface Ports in the IOPLL Reconfig IP Core

Table 16. Avalon-MM Interface Ports in the IOPLL Reconfig IP Core for Intel Stratix 10 Devices

Port	Direction	Description
mgmt_clk	Input	Dynamic reconfiguration clock that drives the IOPLL Reconfig IP core. The maximum input clock frequency is 100 MHz. This clock can be an independent clock source. It must be free running, which means it cannot be connected to the output of the I/O PLL being reconfigured.
mgmt_reset	Input	Active high signal. Synchronous reset input to clear all the data in the IOPLL Reconfig IP core.
mgmt_waitrequest	Output	This port goes high when PLL reconfiguration process started and remains high during PLL reconfiguration. After PLL reconfiguration process completed, this port goes low.
mgmt_write	Input	Active high signal. Asserts to indicate a write operation.
mgmt_read	Input	Active high signal. Asserts to indicate a read operation.
mgmt_writedata[7..0]	Input	Writes data to this port when mgmt_write signal is asserted.
mgmt_readdata[7..0]	Output	Reads data from this port when mgmt_read signal is asserted.
mgmt_address[9..0]	Input	Specifies the address of the data bus for a read or write operation.
reconfig_from_pll[10..0]	Input	Bus that connects to reconfig_from_pll[10..0] bus in the IOPLL Intel FPGA IP core.
reconfig_to_pll[29..0]	Output	Bus that connects to reconfig_to_pll[29..0] bus in the IOPLL IP core.

7.2. Address Bus and Data Bus Settings

Assign a value of "0" for all the unused bits in the address bus and the data bus during reconfiguration operations.

7.2.1. Address Bus and Data Bus Settings for Advanced Mode Reconfiguration

Table 17. Address Bus and Data Bus Settings for Advanced Mode Reconfiguration

Register Name		Address (Binary)	Counter Bit Setting
M Counter	High Count	00000100	• Data[7:0] = high_count • Data[7:0] = low_count • total_count = high_count + low_count
	Low Count	00000111	

continued...

Register Name		Address (Binary)	Counter Bit Setting
	Bypass Enable (10)	00000101	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00000110	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
N Counter	High Count	00000000	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00000010	
	Bypass Enable (10)	00000001	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00000001	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
`C0` Counter	High Count	00011011	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00011110	
	Bypass Enable (10)	00011100	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00011101	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
`C1` Counter	High Count	00011111	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00100010	
	Bypass Enable (10)	00100000	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.

continued...

(10) Perform a read-modify-write operation to configure this setting. PLL may lose lock and can cause reliability issue to your device if you configure with the wrong PLL setting, configure the wrong bit, or overwrite the whole byte for settings that made up just part of one byte.

Register Name		Address (Binary)	Counter Bit Setting
	Odd Division (10)	00100001	• Data[7] = Odd division — Data[7] = 0, odd division is disabled. The selected counter duty cycle = high_count/total_count. — Data[7] = 1, odd division enabled. The selected counter duty cycle = (high_count – 0.5)/total_count.
C2 Counter	High Count	00100011	• Data[7:0] = high_count • Data[7:0] = low_count • total_count = high_count + low_count
	Low Count	00100110	
	Bypass Enable (10)	00100100	• Data[0] = bypass enable — Data[0] = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00100101	• Data[7] = Odd division — Data[7] = 0, odd division is disabled. The selected counter duty cycle = high_count/total_count. — Data[7] = 1, odd division enabled. The selected counter duty cycle = (high_count – 0.5)/total_count.
C3 Counter	High Count	00100111	• Data[7:0] = high_count • Data[7:0] = low_count • total_count = high_count + low_count
	Low Count	00101010	
	Bypass Enable (10)	00101000	• Data[0] = bypass enable — Data[0] = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00101001	• Data[7] = Odd division — Data[7] = 0, odd division is disabled. The selected counter duty cycle = high_count/total_count. — Data[7] = 1, odd division enabled. The selected counter duty cycle = (high_count – 0.5)/total_count.
C4 Counter	High Count	00101011	• Data[7:0] = high_count • Data[7:0] = low_count • total_count = high_count + low_count
	Low Count	00101110	
	Bypass Enable (10)	00101100	• Data[0] = bypass enable — Data[0] = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00101101	• Data[7] = Odd division — Data[7] = 0, odd division is disabled. The selected counter duty cycle = high_count/total_count. — Data[7] = 1, odd division enabled. The selected counter duty cycle = (high_count – 0.5)/total_count.
C5 Counter	High Count	00101111	• Data[7:0] = high_count • Data[7:0] = low_count • total_count = high_count + low_count
	Low Count	00110010	

continued...

Register Name		Address (Binary)	Counter Bit Setting
	Bypass Enable (10)	00110000	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.
	Odd Division (10)	00110001	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
C6 Counter	High Count	00110011	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00110110	
	Bypass Enable (10)	00110100	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1. —
	Odd Division (10)	00110101	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
C7 Counter	High Count	00110111	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00111010	
	Bypass Enable (10)	00111000	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1. —
	Odd Division (10)	00111001	• `Data[7]` = Odd division — `Data[7]` = 0, odd division is disabled. The selected counter duty cycle = `high_count/total_count`. — `Data[7]` = 1, odd division enabled. The selected counter duty cycle = (`high_count` – 0.5)/`total_count`.
C8 Counter	High Count	00111011	• `Data[7:0]` = `high_count` • `Data[7:0]` = `low_count` • `total_count` = `high_count` + `low_count`
	Low Count	00111110	
	Bypass Enable (10)	00111100	• `Data[0]` = bypass enable — `Data[0]` = 1, bypass is enabled. The counter is bypassed with counter division value = 1.

continued...

Register Name		Address (Binary)	Counter Bit Setting
	Odd Division (10)	00111101	• Data[7] = Odd division — Data[7] = 0, odd division is disabled. The selected counter duty cycle = high_count/total_count. — Data[7] = 1, odd division enabled. The selected counter duty cycle = (high_count – 0.5)/total_count.
Charge Pump Current (10)	Charge pump setting [2:0]	00000001	• Data[6:4] = Charge Pump Setting [2:0] — Configure charge pump setting [2:0] on data bit 4 to 6.
	Charge pump setting [5:3]	00001101	• Data[7:5] = Charge Pump Setting [5:3] — Configure charge pump setting [5:3] on data bit 5 to 7.
Bandwidth Setting (10)	—	00001010	• Data[6:3] = Bandwidth Setting — Configure bandwidth setting on data bit 3 to 6.
Ripplecap Setting (10)	—	00001010	• Data[2:1] = Ripplecap Setting — Configure ripplecap setting on data bit 1 and 2.
Calibration (10)	Calibration Request	01001001	• Data[6] = Request Calibration — Data[6] = 1, to request calibration
	Calibration Enable	01001010	• Data[7:0] = Enable Calibration — Data[7:0] = 8'b00000011, to enable calibration

7.2.1.1. Data Bus Setting for Bandwidth Control and Charge Pump

Table 18. Data Bus Setting for Bandwidth Control and Charge Pump (For Low Bandwidth)

Multiply Factor (11)	Low Bandwidth		
	Bandwidth Control Setting Data [6:3]	Charge Pump Setting [2:0] Address = 00000001 Data [6:4]	Charge Pump Setting [5:3] Address = 00001101 Data [7:5]
4–5	4'b0011	3'b011	3'b000
6–15	4'b0011	3'b100	3'b000
16–23	4'b0011	3'b100	3'b000
24–43	4'b0100	3'b100	3'b000
44–64	4'b0101	3'b100	3'b000
65–124	4'b0110	3'b100	3'b000
>124	4'b0110	3'b110	3'b000

(11) If you select the **Use Nondedicated Feedback Path** option under the **Normal** or **Source Synchronous** compensation mode, the Multiply Factor is M x C_i, where M is M counter value while C_i is the Compensated Outclk C counter value. Else, the Multiply Factor is only the M counter value.

Table 19. Data Bus Setting for Bandwidth Control and Charge Pump (For Medium Bandwidth)

Multiply Factor (11)	Medium Bandwidth		
	Bandwidth Control Setting Data [6:3]	Charge Pump Setting [2:0] Address = 00000001 Data [6:4]	Charge Pump Setting [5:3] Address = 00001101 Data [7:5]
4–5	4'b0011	3'b100	3'b000
6–15	4'b0011	3'b111	3'b000
16–23	4'b0011	3'b000	3'001
24–43	4'b0100	3'b000	3'001
44–64	4'b0101	3'b000	3'001
65–124	4'b0101	3'b000	3'001
>124	4'b0110	3'b000	3'001

Table 20. Data Bus Setting for Bandwidth Control and Charge Pump (For High Bandwidth)

Multiply Factor (11)	High Bandwidth		
	Bandwidth Control Setting Data [6:3]	Charge Pump Setting [2:0] Address = 00000001 Data [6:4]	Charge Pump Setting [5:3] Address = 00001101 Data [7:5]
4–5	4'b0011	3'b101	3'b000
6–15	4'b0011	3'b010	3'b001
16–23	4'b0011	3'b100	3'b001
24–43	4'b0100	3'b100	3'b001
44–64	4'b0100	3'b100	3'b001
65–124	4'b0101	3'b100	3'b001
>124	4'b0101	3'b100	3'b001

7.2.1.2. Data Bus Setting for Ripplecap

Table 21. Data Bus Setting for Ripplecap

Multiply Factor (12)	Ripplecap Setting Address = 00001010 Data [2:1]
4-5	2'b00
6-15	2'b00
16-23	2'b10

continued...

(12) If you select the **Use Nondedicated Feedback Path** option under the **Normal** or **Source Synchronous** compensation mode, the Multiply Factor is M x C_i, where M is M counter value while C_i is the Compensated Outclk C counter value. Else, the Multiply Factor is only the M counter value.

Multiply Factor (12)	Ripplecap Setting Address = 00001010 Data [2:1]
24-43	2'b00
44-64	2'b10
65-124	2'b10
>124	2'b10

7.2.2. Output Clock and the Corresponding Data Bit Setting for Clock Gating Reconfiguration

Table 22. Output Clock and the Corresponding Data Bit Setting for Clock Gating Reconfiguration

Output Clock	Data Bus Bit Setting (Binary)	
C0	data[0]	Gated = 1'b0 Ungated = 1'b1
C1	data[1]	
C2	data[2]	
C3	data[3]	
C4	data[4]	
C5	data[5]	
C6	data[6]	
C7	data[7]	

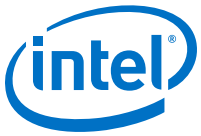
7.2.3. Data Bus Setting for Dynamic Phase Shift for IOPLL Reconfig IP Core

Table 23. Data Bus Setting for Dynamic Phase Shift for IOPLL Reconfig IP Core

Write Data Bus Setting	Description	
data[2:0]	Determines the number of phase shifts per dynamic phase shift operation. Up to seven phase shifts per operation are possible. Each phase shift step is equal to 1/8 of I/O PLL VCO period.	
data[3]	Determines the direction of dynamic phase shift. When data[3] = 0, phase shift is in negative direction. When data[3] = 1, phase shift is in positive direction.	
data[7:4]	Determines the counter to be selected to perform dynamic phase shift operation.	
	Counter Name	data[7:4]
	C0	4'b0000

continued...

(12) If you select the **Use Nondedicated Feedback Path** option under the **Normal** or **Source Synchronous** compensation mode, the Multiply Factor is M x C_i, where M is M counter value while C_i is the Compensated Outclk C counter value. Else, the Multiply Factor is only the M counter value.



Write Data Bus Setting	Description	
	Counter Name	**data[7:4]**
	C1	4'b0001
	C2	4'b0010
	C3	4'b0011
	C4	4'b0100
	C5	4'b0101
	C6	4'b0110
	C7	4'b0111
	C8	4'b1000
	All C counters	4'b1111

8. Intel Stratix 10 Clocking and PLL User Guide Archives

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
18.1	Intel Stratix 10 Clocking and PLL User Guide
18.0	Intel Stratix 10 Clocking and PLL User Guide
17.1	Intel Stratix 10 Clocking and PLL User Guide

9. Document Revision History for the Intel Stratix 10 Clocking and PLL User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.07.01	19.2	• Updated the *Intel Stratix 10 Clock Input Pins Resources* table. — Added Intel Stratix 10 devices: TX 850, TX 1100, GX 1660, and GX 2110. — Removed unsupported Intel Stratix 10 devices: MX 1100, GX 4500, GX 5500, SX 4500, and SX 5500. — Updated the number of resources available for Intel Stratix 10 TX 1650 and TX 2100. • Updated the description in the *Root Clock Gate* section. • Added new guideline topics: — *Guideline: I/O PLL Jitter Performance* — *Guideline: Clock Gating*
2018.12.24	18.1	• Updated the *I/O PLL High-Level Block Diagram for Intel Stratix 10 Devices*. • Updated the *Power-Up Calibration* section. • Added the constraints for IOPLL IP core in the *IP Core Constraints* section. • Added information on advanced mode reconfiguration in the following sections: — *Guideline: I/O PLL Reconfiguration* — *IOPLL Reconfig Intel Stratix 10 FPGA IP Core* — *IOPLL Reconfig IP Core Reconfiguration Modes* • Updated the steps to perform I/O PLL reconfiguration using advanced mode in the *Advanced Mode Reconfiguration* section. • Added *Design Example 2: Advanced Mode Reconfiguration Using IOPLL Reconfig IP Core*.
2018.08.03	18.0	Added diagrams on PLL cascading connectivity in the *PLL Cascading* section.
2018.05.07	18.0	• Added a link to the *Design Recommendations User Guide* for details on clock assignments in the Intel Quartus Prime software. • Clarified that the dynamic clock switchover can be optionally made glitch free using additional external soft logic. • Added note about the non-dedicated feedback path for the source synchronous compensation mode and normal compensation mode in the *PLL Features in Intel Stratix 10 Devices* table. • Added note about the non-dedicated feedback path for the source synchronous compensation mode and normal compensation mode in the *I/O PLL High-Level Block Diagram for Intel Stratix 10 Devices* diagram. • Updated the core clock compensation methods in the *Clock Feedback Modes* section. • Updated the *Direct Compensation Mode* section. • Added information on non-dedicated feedback path in normal or source synchronous compensation mode in the *Clock Multiplication and Division* section.

continued...

Document Version	Intel Quartus Prime Version	Changes
		• Updated the *PLL Cascading* section. • Updated the description in the following sections to mention that the `mgmt_clk` and `scanclk` signals must be free running. — *Guideline: I/O PLL Reconfiguration* — *Avalon-MM Interface Ports in the IOPLL Reconfig IP Core for Intel Stratix 10 Devices* • Added information on advanced mode configuration. — Added the advanced mode configuration in the *IOPLL Reconfig IP Core Reconfiguration Modes* table. — Added new section: *Advanced Mode Reconfiguration*. — Added new table: *Address Bus and Data Bus Settings for Advanced Mode Reconfiguration*. — Added new table: *Data Bus Setting for Bandwidth Control and Charge Pump*. — Added new table: *Data Bus Setting for Ripplecap*. • Added a note on recalibration for .mif streaming reconfiguration in the *.mif Streaming Reconfiguration* section. • Corrected step 3 in the *Dynamic Phase Shift Reconfiguration* section. • Corrected the waveforms for `mgmt_address[7:0]` and `mgmt_writedata[7:0]` in the *Waveform Example for .mif Streaming Reconfiguration Design Example* diagram. • Removed the dynamic phase shift feature for fPLL in the following sections: — Removed phase shift resolution for fPLL and updated the note to phase shift resolution in the *PLL Features in Intel Stratix 10 Devices* table. — Removed description on fPLL in the *Programmable Phase Shift* section. — Removed description on fPLL in the *PLL Reconfiguration and Dynamic Phase Shift* section. • Added **Compensated Outclk** and **Use Nondedicated Feedback Path** parameters in the *IOPLL IP Core Parameters - PLL Tab for Intel Stratix 10 Devices* table. • Added **Create a permit_cal signal to connect with an upstream PLL** parameter in the *IOPLL IP Core Parameters - Cascading Tab* table. • Added `permit_cal` port in the *IOPLL Ports for Intel Stratix 10 Devices* table. • Renamed the following IP cores as per Intel rebranding: — Renamed Intel FPGA IOPLL Reconfig IP core to IOPLL Reconfig Intel FPGA IP core. — Renamed Intel FPGA IOPLL IP core to IOPLL Intel FPGA IP core. — Renamed Stratix 10 Clock Control IP core to Clock Control Intel Stratix 10 FPGA IP core.

Date	Version	Changes
December 2017	2017.12.07	• Updated the *Dedicated Clock Resources Within a Clock Sector* diagram. • Updated description in the *Programmable Clock Routing* section. • Updated *Intel Stratix 10 Clock Input Pins Resources* table. — Added resources for Intel Stratix 10 TX and MX devices. — Updated resources for the following devices: • GX 1650 • GX 2100 • SX 1650 • SX 2100 • GX 2500 • GX 2800 • SX 2500 • SX 2800 • Added note to core signals in *Intel Stratix 10 Programmable Clock Routing Resources* table. • Updated *Clock Gating and Clock Divider in Intel Stratix 10 Clock Network* diagram. • Added links and updated description in the *Root Clock Gate* section. • Added links and updated description in the *Sector Clock Gate* section. • Updated the *Clock Gating Timing Diagram*. • Updated description in the *Clock Divider* section. • Updated *PLL Features in Intel Stratix 10 Devices* table. — Updated C counter divide factors for I/O PLL. — Updated the note to phase shift resolution and updated the phase shift resolution for fPLL. • Updated the *Reset* section. — Updated the note about the conditions to reset the I/O PLL. — Removed description on fPLL reset signal (`pll_powerdown`). • Updated the description in the following sections. — *Clock Feedback Modes* — *Direct Compensation Mode* — *Source Synchronous Compensation Mode* — *Normal Compensation Mode* • Updated the description in the *PLL Cascading* section. • Added a requirement for automatic clock switchover mode. • Updated description in the *Manual Clock Switchover* section. • Removed the guidelines on PLL reconfiguration using `.mif` streaming in the *Guideline: Configuration Constraints* section. • Added design examples for IOPLL and IOPLL Reconfig IP cores. • Updated port names in the *Connectivity between the IOPLL and IOPLL Reconfig IP Cores in the Intel Quartus Prime Software* diagram. • Updated `reconfig_from_pll[9..0]` to `reconfig_from_pll[10..0]` in the following sections: — *Connectivity between the IOPLL and IOPLL Reconfig IP Cores in the Intel Quartus Prime Software* diagram — *Connecting the IOPLL and IOPLL Reconfig IP Cores* section — *Avalon-MM Interface Ports in the IOPLL Reconfig IP Core* section • Added a note to the *IOPLL Reconfig IP Core Reconfiguration Modes* table. • Updated the *Clock Control IP Core Parameters for Intel Stratix 10 Devices* table. — Updated **Ensure glitch free clock switchover** description. — Updated **Clock Enable Type** description. — Updated **Enable Register Mode** value and description. • Updated **Multiply Factor (M-Counter)** legal value in the *IOPLL IP Core Parameters - PLL Tab for Intel Stratix 10 Devices* table.

continued...

Date	Version	Changes
		• Updated *IOPLL IP Core Parameters - Settings Tab for Intel Stratix 10 Devices* table. — Updated parameter from **Enable access to PLL LVDS_CLK/ LOADEN output port** to **Access to PLL LVDS_CLK/LOADEN output port**, the legal value, and description. • Updated `extswitch` description in the *IOPLL Ports for Intel Stratix 10 Devices* table. • Updated `updn` description in the *Dynamic Phase Shift Ports in the IOPLL IP Core* table. • Updated descriptions for `data[3]` and `data[7:4]` in the *Data Bus Setting for Dynamic Phase Shift for IOPLL Reconfig IP Core* table. • Updated the following terms: — Changed LogicLock Plus to Logic Lock — Changed TimeQuest Timing Analyzer to Timing Analyzer • Updated the following IP names: — Changed Altera IOPLL to IOPLL — Changed Altera IOPLL Reconfig to IOPLL Reconfig — Changed Altera In-System Sources & Probe to In-System Sources & Probes
May 2017	2017.05.26	• Updated the following sections: — Clock Sector — Programmable Clock Routing — Internal Logic — Zero-Delay Buffer Mode — External Feedback Mode — User Calibration • Updated the default feedback mode for normal and source synchronous compensation modes. • Updated scale factor for Post-Scale Counter, L in Clock Multiplication and Division section. • Updated minimum phase shift increment for fPLL in the following sections: — Programmable Phase Shift — PLL Reconfiguration and Dynamic Phase Shift • Changed CLKUSR to OSC_CLK_1 in PLL Calibration section. • Updated IOPLL IP core. • Added Intel Stratix 10 Clocking and PLL Design Considerations chapter. • Added IOPLL Reconfig IP core.
October 2016	2016.10.31	Initial release.